

The Geochemist's Workbench®

Release 17

GWB
Reactive Transport
Modeling Guide

The Geochemist's Workbench®

Release 17

GWB

Reactive Transport

Modeling Guide

Craig M. Bethke

Brian Farrell

Melika Sharifi

Aqueous Solutions, LLC

Champaign, Illinois

Printed October 9, 2023

This document © Copyright 2023 by Aqueous Solutions LLC. All rights reserved. Earlier editions copyright 2000–2022. This document may be reproduced freely to support any licensed use of the GWB software package.

Software copyright notice: Programs GSS, Rxn, Act2, Tact, SpecE8, Gtplot, TEdit, React, Phase2, P2plot, X1t, X2t, Xtplot, and ChemPlugin © Copyright 1983–2023 by Aqueous Solutions LLC. An unpublished work distributed via trade secrecy license. All rights reserved under the copyright laws.

The Geochemist's Workbench®, ChemPlugin™, We put bugs in our software™, and The Geochemist's Spreadsheet™ are a registered trademark and trademarks of Aqueous Solutions LLC; Microsoft®, MS®, Windows 11®, and Windows 10® are registered trademarks of Microsoft Corporation; PostScript® is a registered trademark of Adobe Systems, Inc. Other products mentioned in this document are identified by the trademarks of their respective companies; the authors disclaim responsibility for specifying which marks are owned by which companies. The software uses zlib © 1995–2005 Jean-Loup Gailly and Mark Adler, and Expat © 1998–2006 Thai Open Source Center Ltd. and Clark Cooper.

The GWB software was originally developed by the students, staff, and faculty of the Hydrogeology Program in the Department of Geology at the University of Illinois Urbana-Champaign. The package is currently developed and maintained by Aqueous Solutions LLC.

Address inquiries to:

Aqueous Solutions LLC
301 North Neil Street, Suite 400
Champaign, IL 61820 USA

Warranty: The Aqueous Solutions LLC warrants only that it has the right to convey license to the GWB software. Aqueous Solutions makes no other warranties, express or implied, with respect to the licensed software and/or associated written documentation. Aqueous Solutions disclaims any express or implied warranties of merchantability, fitness for a particular purpose, and non-infringement. Aqueous Solutions does not warrant, guarantee, or make any representations regarding the use, or the results of the use, of the Licensed Software or documentation in terms of correctness, accuracy, reliability, currentness, or otherwise. Aqueous Solutions shall not be liable for any direct, indirect, consequential, or incidental damages (including damages for loss of profits, business interruption, loss of business information, and the like) arising out of any claim by Licensee or a third party regarding the use of or inability to use Licensed Software. The entire risk as to the results and performance of Licensed Software is assumed by the Licensee. See License Agreement for complete details.

License Agreement: Use of the GWB is subject to the terms of the accompanying License Agreement. Please refer to that Agreement for details.

Cover photo: Salinas de Janubio by Jorg Hackemann

Contents

Chapter List

Contents	v
Detailed Contents.....	vii
Reactive Transport Modeling with the GWB.....	1
Modeling Overview	3
Using X1t.....	47
Using X2t.....	113
Cluster Computing	153
Using Xtplot	157
Appendix: Heterogeneity.....	169
Appendix: Importing from MODFLOW	179
Appendix: Further Reading.....	185

Detailed Contents

Contents	v
Detailed Contents.....	vii
Reactive Transport Modeling with the GWB.....	1
1.1 Programs in the Xt package	1
1.2 Before you start.....	2
Modeling Overview	3
2.1 Capabilities of the software	3
2.2 Conceptual model.....	4
2.3 Simulation procedure	5
2.4 Discretization	6
2.5 Node indexing	7
2.6 Field variables	9
2.7 Initial conditions	13
2.8 Boundary fluids	14
2.9 Boundary conditions.....	17
2.10 Reaction intervals	18
2.11 Kinetic reactions and gas buffers	22
2.12 Porosity evolution	23
2.13 Permeability correlation	24
2.14 Groundwater flow	26
2.15 Polythermal simulations.....	27
2.16 Internal heat production.....	29
2.17 Dual porosity model.....	30
2.18 Mobile colloids	36
2.19 Stable isotope transport	39
2.20 Time marching	41
2.21 Multicore execution.....	42
2.22 Running a model	43
Using X1t.....	47
3.1 Defining the domain.....	47
3.2 Setting flow rate	52

3.3 Mass transport	56
3.4 Heat transfer	58
3.5 Example input files	59
3.6 Example: rainwater infiltrating a quartz aquifer.....	60
3.7 Example: quartz precipitation in a vein	64
3.8 Example: weathering in a soil profile	69
3.9 Example: Pb contamination	77
3.10 Example: groundwater chromatography	84
3.11 Example: steam flood	89
3.12 Example: dual porosity model	95
3.13 Example: mobile colloid	101
3.14 Example: stable isotope transport.....	106
3.15 X1t command line	112
Using X2t	113
4.1 Defining the domain	113
4.2 Wells	119
4.3 Calculating the flow field	125
4.4 Importing the flow field	129
4.5 Importing from MODFLOW	131
4.6 Mass transport	132
4.7 Heat transfer	135
4.8 Example input files	135
4.9 Example: metals contamination of an aquifer	136
4.10 Example: steam flood	142
4.11 X2t command line	151
Cluster Computing	153
5.1 Working on a cluster	153
5.2 Running a simulation.....	154
5.2.1 Running a batch job	154
5.2.3 Interactive execution	155
Using Xtplot.....	157
6.1 Map View	158
6.2 XY Plot configuration	160
6.3 Plot types	162
6.4 Editing plot appearance	163
6.5 Scatter data.....	163
6.6 Loading and saving plot configuration.....	164
6.7 Exporting the plot	165

6.8 Xtpplot command line.....	167
Appendix: Heterogeneity.....	169
A.1 Values and random fields	170
A.2 Table datasets	171
A.3 In-line tables.....	172
A.4 Simple expressions	172
A.5 Basic scripts.....	174
A.6 Compiled functions.....	175
Appendix: Importing from MODFLOW	179
B.1 Discretization and budget files.....	180
B.2 Importing to and running X2t.....	180
B.3 MODFLOW log file	181
B.4 Examples	181
Appendix: Further Reading.....	185
C.1 Reactive transport	185
C.2 Fluid viscosity.....	186
C.3 Silicate kinetics and weathering	186
C.4 Steam flooding.....	186
C.5 MODFLOW-2000	187

Reactive Transport Modeling with the **GWB**

The Xt software package included in **The Geochemist's Workbench Professional** release calculates numerical models of groundwater transport within reacting geochemical systems in one and two dimensions, and renders the calculation results graphically. Versions of the programs for use on a computing cluster are distributed with the **GWB Advanced Professional** release.

Models of this class are known as reactive transport models. A reactive transport model is a groundwater flow and transport model coupled to a chemical reaction model. The transport calculation accounts for the movement of groundwater and the chemical species dissolved in it by advection, hydrodynamic dispersion, and molecular diffusion. The transport calculation can also account for the transfer of heat through the subsurface by advection and conduction.

The chemical reaction model is very similar to the **React** modeling program in The Geochemist's Workbench, as described in the ***GWB Reaction Modeling Guide***.

1.1 Programs in the Xt package

The Xt package is composed of three programs:

- **X1t**, used to model reactive transport in one linear, radial, or spherical dimension
- **X2t**, used to model reactive transport in two dimensions
- **Xtplot**, which renders the results of **X1t** and **X2t** simulations graphically in map view, and in terms of xy plots

Programs **X1t** and **X2t** are available for use on computing clusters. The cluster versions are named, respectively, **cX1t** and **cX2t**.

In addition to information contained in this guide, the ***GWB Command Reference*** contains details of the various commands you can use to configure **X1t** and **X2t**.

1.2 Before you start

Before beginning to work with the Xt programs, it is important that you have a full understanding of the geochemical modeling techniques used in the **React** program in the GWB. It is advisable, furthermore, to begin a project by using **React** to construct a reaction model of your geochemical system.

Once you are satisfied with the reaction model, you can begin with confidence to construct a coupled model of reaction and transport.

This manual is intended to document how the Xt package can be applied to model reactive transport; it is an adjunct to the ***GWB Essentials Guide*** and the ***GWB Reaction Modeling Guide***. This manual is not intended to serve as a general reference on reactive transport modeling. An appendix to this guide, [Further Reading](#), contains a number of literature citations that provide a starting point for learning more about this class of models.

Modeling Overview

This chapter gives an overview of how you can use programs **X1t**, **X2t**, and **Xtplot** in the Xt package to model reactive transport in porous media. Subsequent chapters discuss the specifics of using **X1t** to construct reactive transport models in one dimension, and **X2t** to trace simulations in two dimensions. You should read this chapter first, before beginning the later chapters.

2.1 Capabilities of the software

Reactive transport models, as the name suggests, are composed of two elements: a transport calculation and a reaction model. The transport calculation accounts for the movement of groundwater and the transport of chemical species dissolved in it by

- advection, the movement of groundwater through the subsurface
- hydrodynamic dispersion, the mechanical mixing within the groundwater flow
- molecular diffusion

The transport calculation can also account for the transfer of heat through the subsurface by advection and conduction.

The reaction model is broadly the same as that implemented in **React** in the GWB, as described in the ***GWB Reaction Modeling Guide***. Reaction modeling capabilities include

- distribution of dissolved mass among aqueous species
- redox equilibrium and disequilibrium
- Debye-Hückel variations, SIT, and “Pitzer” (HMW) activity models
- gas buffering
- minerals and solid solution end members held in local equilibrium with fluid, as well as those that dissolve and precipitate according to kinetic rate laws
- sorption onto mineral surfaces and surface complexation
- flexible specification of kinetic rate laws, including catalysis, inhibition, nucleation, and nonlinear effects
- the effects of microbial metabolism and growth
- the fractionation of stable isotopes

4 *GWB Reactive Transport Modeling*

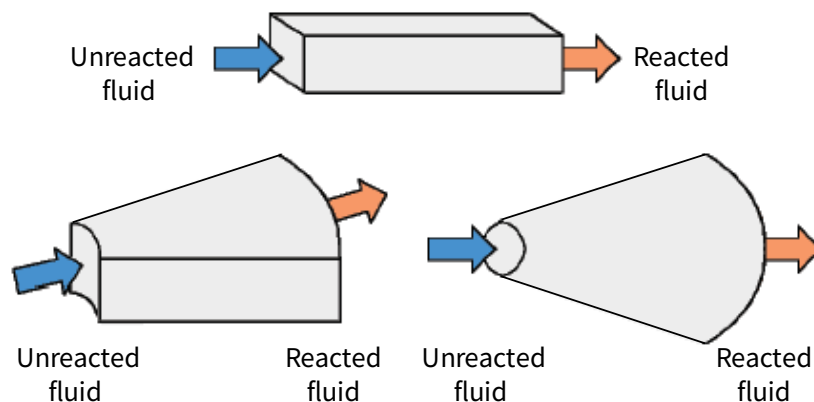
You can use **X1t** and **X2t** to construct models in which

- fluid migrates at fixed rates and patterns, or in rates and patterns that reflect the evolving permeability of the medium
- permeability varies over the simulation as porosity and the mineral composition of domain change
- the medium is isothermal or polythermal
- the domain and initial conditions are homogeneous or heterogeneous
- the domain is Cartesian in one or two dimensions, radial or spherical in one dimension, or axisymmetric in two dimensions
- mobile colloids transport sorbed mass through the domain
- isotopic mass fractionates as it is transported through the domain

The package includes the module **Xtplot**, which is similar to **Gtplot** and **P2plot**, for rendering calculation results graphically.

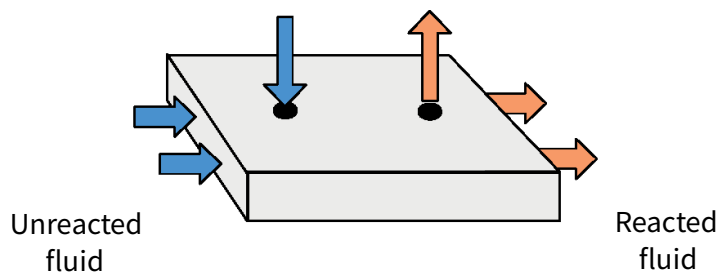
2.2 Conceptual model

In an **X1t** simulation, you set the initial mineralogy of the medium and the chemical composition of the pore fluid. Over the course of the simulation, an unreacted fluid of a composition you specify enters either end of the domain and reacts as it passes along a linear, radial, or spherical coordinate.



The flow of fluid along the domain displaces reacted fluid from the opposite end.

An **X2t** simulation is similar, except that flow occurs in a two-dimensional rectilinear or axisymmetric coordinate, and fluid can enter and leave the domain at wells.



You specify whether the boundaries of the domain are open or closed to flow, and optionally the fluid discharge across them, and either the rate of flow into or out of each well, or head at the well. As with **X1t**, you set the initial conditions within the domain, as well as the composition of any fluids flowing across boundaries or injected into wells.

2.3 Simulation procedure

X1t and **X2t** are “time marching” programs: you set the domain’s initial condition and constrain the composition of the fluids that move into it. The program predicts how the domain changes over time as it reacts with the migrating fluid.

In tracing a simulation, **X1t** and **X2t** proceed in the following manner:

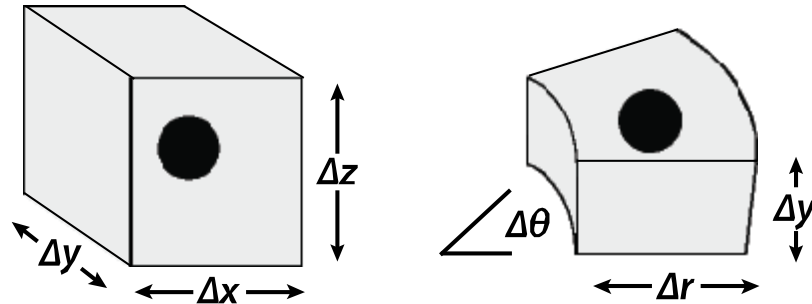
- The model discretizes the domain by dividing it into nodal blocks, and sets the physical properties, such as porosity and permeability, at each block.
- It figures the initial chemical state of each nodal block and of whatever fluids pass into the domain across its boundaries, or are injected into wells.
- It marches through time in discrete time steps, Δt , until it reaches the simulation’s ending time. In taking a time step, the program figures the groundwater flow field and calculates how transport over the time step affects the chemical composition and, optionally, temperature of each nodal block over the time step. It then determines the chemical state of each nodal block corresponding to the revised composition and temperature, and reflecting the progress of any kinetic reactions.

The time marching algorithm employed, in which the model at each time step evaluates the transport equations separately from the chemical model, is known in the field of reactive transport modeling as the “operator splitting method”.

2.4 Discretization

To undertake a reactive transport simulation, programs **X1t** and **X2t** automatically divide the modeling domain into nodal blocks. This process is known as *discretization* because it separates the domain into discrete parts.

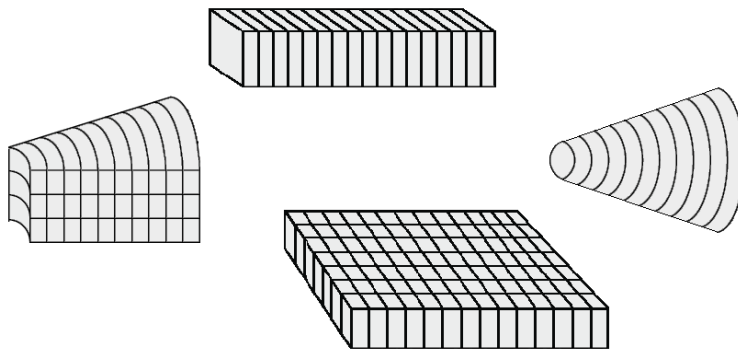
Nodal blocks look like



Each nodal block has fixed dimensions Δx , Δy , and Δz in a linear domain, Δr , $\Delta \theta$, and Δy in a radial domain, or Δr and $\Delta \theta$ in a spherical domain. A nodal point is located at the center of each block, as shown.

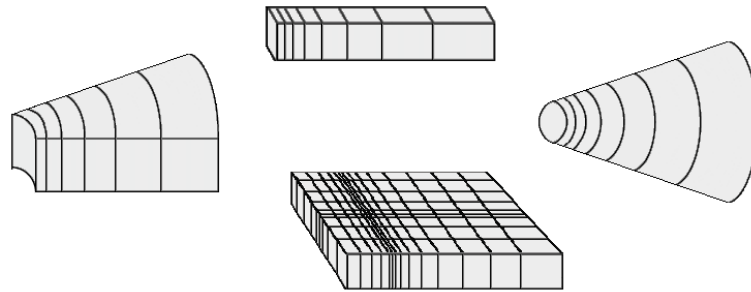
The properties of the entire nodal block are projected onto the nodal point, so the nodal block carries a single value for porosity, permeability, temperature, pH, and so on. The nodal blocks and points may be referred to simply as “nodes”.

The nodal blocks fit together to form the domain over which the transport equations are solved.



In the examples above, the nodes are spaced evenly. In other words, the dimensions Δx , Δy , Δz (or Δr , $\Delta \theta$, Δy) of each block are the same.

It is also possible to set a simulation in which node sizes vary



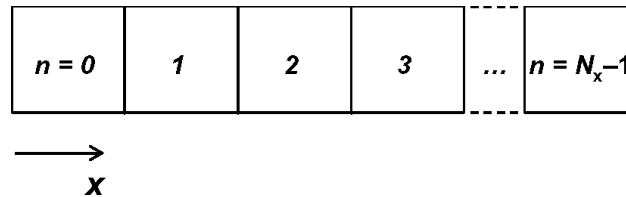
across the domain, as described in the next two chapters.

X1t and **X2t** solve the finite difference equations representing groundwater flow, mass transport, and heat transfer across the discretized domain using the forward-in-time, upstream-weighted method. The programs solve for the chemical state of the system in the same manner as **React**.

2.5 Node indexing

X1t and **X2t** are written in the C++ programming language and follow the C++ convention for indexing nodes. In this convention, the index for N entities varies from 0 to $N - 1$, rather than from 1 to N . An **Xt** domain has N_x columns of nodes and N_y rows (N_y is always 1 in **X1t** simulations, which are one-dimensional), so there are a total of $N = N_x \times N_y$ nodes.

A domain in **X1t** has nodes numbered 0 through $N_x - 1$.

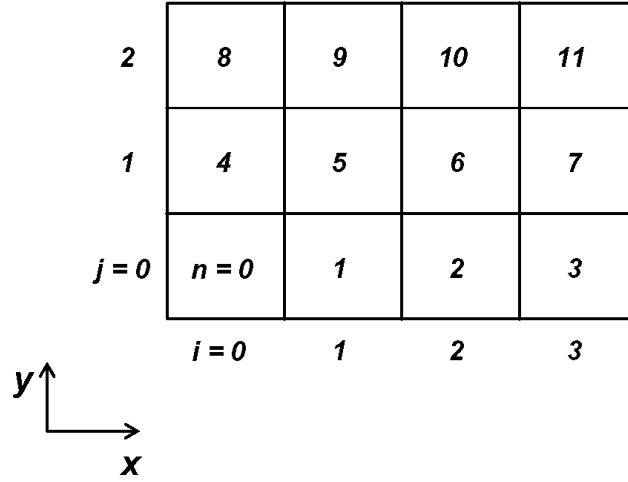


In **X2t**, the node in the lower left-hand corner of the domain is number $n = 0$, the next number $n = 1$, and so on, until the upper right-hand corner, which is number $n = N - 1$.

8 GWB Reactive Transport Modeling

A 4×3 domain, then, would be indexed

2	8	9	10	11
1	4	5	6	7
$j = 0$	$n = 0$	1	2	3
	$i = 0$	1	2	3



In general, columns of nodes are indexed $i = 0$ to $N_x - 1$, and rows from $j = 0$ to $N_y - 1$. The index n of a node at column i and row j is given by the equation

$$n = j \times N_x + i \quad (2.1)$$

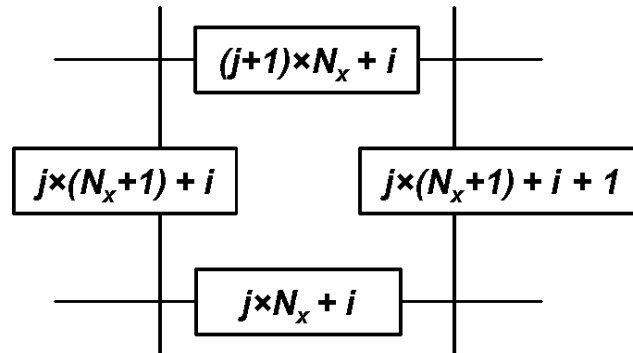
Conversely, the i, j indices of node number n can be calculated as

$$i = n \% N_x \quad (2.2)$$

$$j = n / N_x \quad (2.3)$$

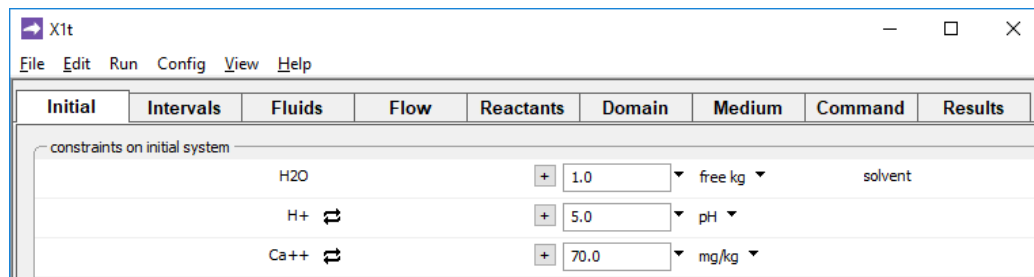
where % is the modulus operator.

On occasion, such as when specifying the groundwater discharge from one node to another, it is necessary to refer to the boundaries between nodal blocks. The boundaries separating node i, j from its neighbors are indexed according to the scheme



2.6 Field variables

In constructing a reactive transport model, you need to set the initial properties of the medium and the chemistry of the initial fluid at each node in the domain. For an initially homogeneous domain, this is easily accomplished using the GUI by setting a variable to a specific value within the appropriate pane or dialog box. On the **Initial** and **Fluids** panes, for example,



you can set the chemical composition of the initial system and boundary fluids, respectively.

10 *GWB Reactive Transport Modeling*

On the **Medium** pane,

The screenshot shows the X1t software interface with the 'Medium' pane selected. The pane is divided into three main sections: 'medium properties', 'mass transport', and 'permeability'. Each section contains several input fields with numerical values and units, and a small '+' icon for each field. The 'medium properties' section includes fields for diffusion coefficient (1.0e-06 cm2/s), porosity (0.3), inert volume (0.0 cm3), thermal conductivity (0.004 cal/cm/s/C), heat capacity (cpw (fluid) 1.0 cal/g/C, cpr (minerals) 0.2 cal/g/C), internal heat source (temperature: minimum and maximum in °C, and a value of 0.0 cal/cm3/s), and longitudinal dispersivity (200.0 cm). The 'mass transport' section includes a field for permeability: log kx (darcy) = (A x porosity (volume fraction)) + B. The 'permeability' section includes fields for A (porosity) (15.0) and B (intercept) (-5.0 darcy). At the bottom of the pane are 'add' and 'delete' buttons.

you can set various medium properties. Alternatively, you can use commands such as


```
pH = 5
Ca++ = 70 mg/kg
porosity = 30%
dispersivity = 200 cm
```

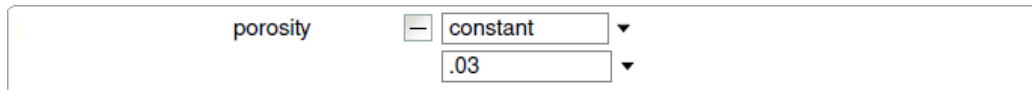
To define properties within a heterogeneous medium, in contrast, you need to set a value at each individual node. In X1t, properties that may be set to be heterogeneous over the domain are known as *field variables*. You can use field variables to define:

- the medium properties, including porosity, permeability, dispersivity, and so on
- the initial conditions, including temperature, fluid chemistry, mineral mass, biomass, and rate of internal heat production
- the reactant masses and the target values for sliding activity and fugacity buffers
- the groundwater flow field
- the lengths (Δx) and widths (Δy) of nodal blocks

X1t and **X2t** provide a number of ways to set a field variable. They may be entered as

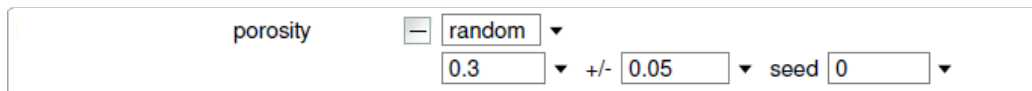
- constant values, in which case the variable is homogeneous
- values randomly chosen within a Gaussian distribution, as defined by a mean value and standard deviation about the mean
- the result of evaluating an expression
- values taken from a dataset of numbers in tabular form
- an “inline table” entered directly from the command line or GUI
- the result of evaluating a script or compiled function you supply

To set the initial porosity in any of the above-mentioned ways using the GUI, click on the **Medium** pane and then on the  symbol next to “porosity”. The following examples show porosity set to a constant value,



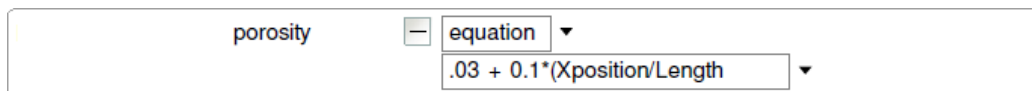
The GUI shows a panel for 'porosity'. On the left is a minus sign icon. To its right is a dropdown menu with 'constant' selected. Below this dropdown is a text input field containing '.03'.

as a random value,



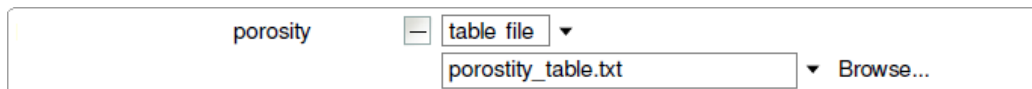
The GUI shows a panel for 'porosity'. On the left is a minus sign icon. To its right is a dropdown menu with 'random' selected. Below this dropdown is a text input field containing '0.3'. To the right of this field is a dropdown menu with '+/-' selected, followed by a text input field containing '0.05'. To the right of this is the text 'seed' followed by a dropdown menu with '0' selected.

as a simple expression to be evaluated,



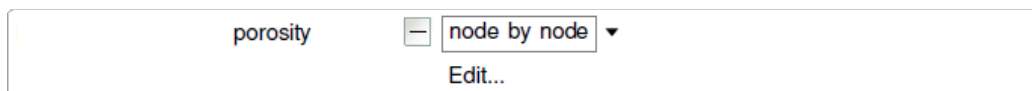
The GUI shows a panel for 'porosity'. On the left is a minus sign icon. To its right is a dropdown menu with 'equation' selected. Below this dropdown is a text input field containing the expression '.03 + 0.1*(Xposition/Length)'.

to be taken from a tabulated dataset of numbers,



The GUI shows a panel for 'porosity'. On the left is a minus sign icon. To its right is a dropdown menu with 'table file' selected. Below this dropdown is a text input field containing 'porosity_table.txt'. To the right of this field is a dropdown menu with 'Browse...' selected.

and set directly within a node-by-node editor.



The GUI shows a panel for 'porosity'. On the left is a minus sign icon. To its right is a dropdown menu with 'node by node' selected. Below this dropdown is a text input field containing 'Edit...'.

12 *GWB Reactive Transport Modeling*

	0	1	2	3	4	5
0	0.26	0.28	0.3	0.32	0.32	0.32

You could alternatively specify porosity in any of these ways using one of the following commands

```
porosity = 0.3
porosity = 0.3 +/- 0.05
porosity eqn = "0.3 + 0.1*(Xposition/Length) "
porosity = porosity_table.txt
porosity = { .26 .28 .30 .32 .32 .32 }
```

A number of properties—the internal heat source or the rate constants for kinetic reactions, for example—can be set to change with time over the course of the simulation. The programs evaluate such transient field variables at the individual nodes in the domain not only initially, but upon undertaking each time step over the course of a run.

A field variable, by default, holds steady throughout a run, but you can specify transient behavior by checking the ☒ transient box in the GUI next to where the variable is set. The example below

internal heat source temperature: minimum °C maximum °C

☐ ☒ equation cal/cm3/s

780 * EXP(-3.2e-10 * Time) ☒ transient

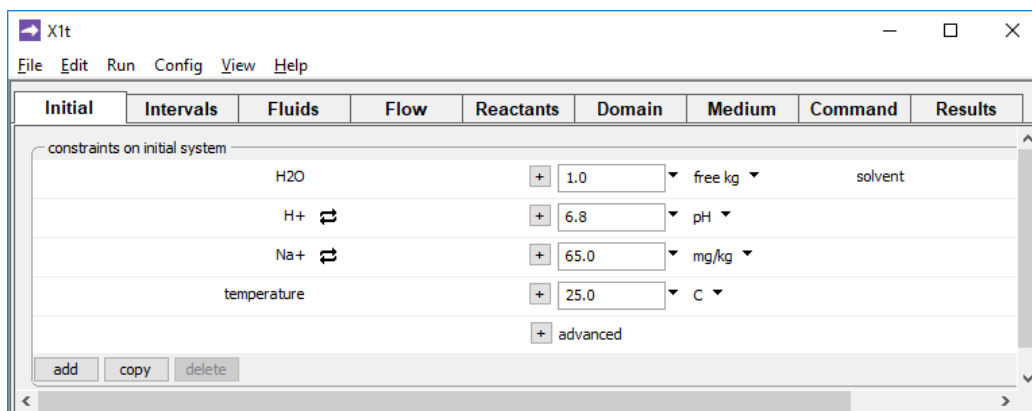
sets a heat source that decays in the run exponentially with time. You can also prescribe transient behavior on the command line with the `transient` keyword. For example,

```
heat_source = eqn "780 * EXP(-3.2e-10 * Time)" transient
```

The options for setting field variables are described in detail in the [Heterogeneity](#) appendix to this guide.

2.7 Initial conditions

X1t and **X2t** calculate the initial chemical conditions at each node in the domain, according to the constraints you supply. You constrain the initial conditions on the **Initial** pane,



where you set a basis vector as if you were configuring a **SpecE8** or **React** model.

As with the other apps, you can alternatively move to the **Command** pane and enter commands there,

```
pH = 6.8
Na+ = 65 mg/kg
```


and so on. When using the command line interface, the programs maintain a scope, which is the target to which the commands apply. The scope, by default, is global, which means the commands you entered above would apply not only to the initial conditions, but also to whatever boundary fluids exist in the configuration, or are created later.

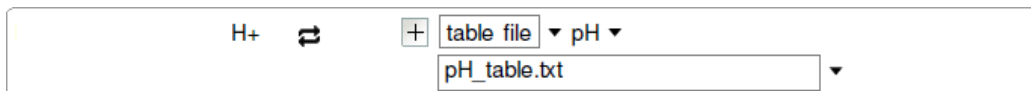
To constrain only the initial conditions, use the `scope` command to set the scope to `initial` before you enter the basis constraints:

```
scope initial
  pH = 6.8
  Na+ = 65 mg/kg
scope
```

Entering a `scope` command without an argument, as shown above, returns the scope to global, as does entering a command of global significance, such as `kinetic` or `length`.

It is good practice to close a scope block, as we have here, rather than leaving a dangling setting.

Unlike **SpecE8** or **React**, **X1t** and **X2t** allow you to set constraints on basis entries as field variables. You can, for example, set the initial pH to vary across the domain. To set a heterogeneous constraint, click on the  symbol for the basis entry and choose one of the options for setting a field variable. The option below



constrains a basis entry as a field variable. In this case, the initial pH would vary over the domain, according to the data in dataset “pH_table.txt”. Working from the command line,

```
scope initial
  pH = pH_table.txt
scope
```

would set up the same pH field.

When a basis entry is constrained by equilibrium with a mineral, you need to set the amount of the mineral found within a nodal block. You can do so directly, in units such as moles, grams, or cm³, as you might in **SpecE8** or **React**. In **X1t** and **X2t**, however, you should constrain the amount of a mineral relative to fluid mass within a nodal block, or the block’s bulk volume, in units such as mmol/kg or volume%.

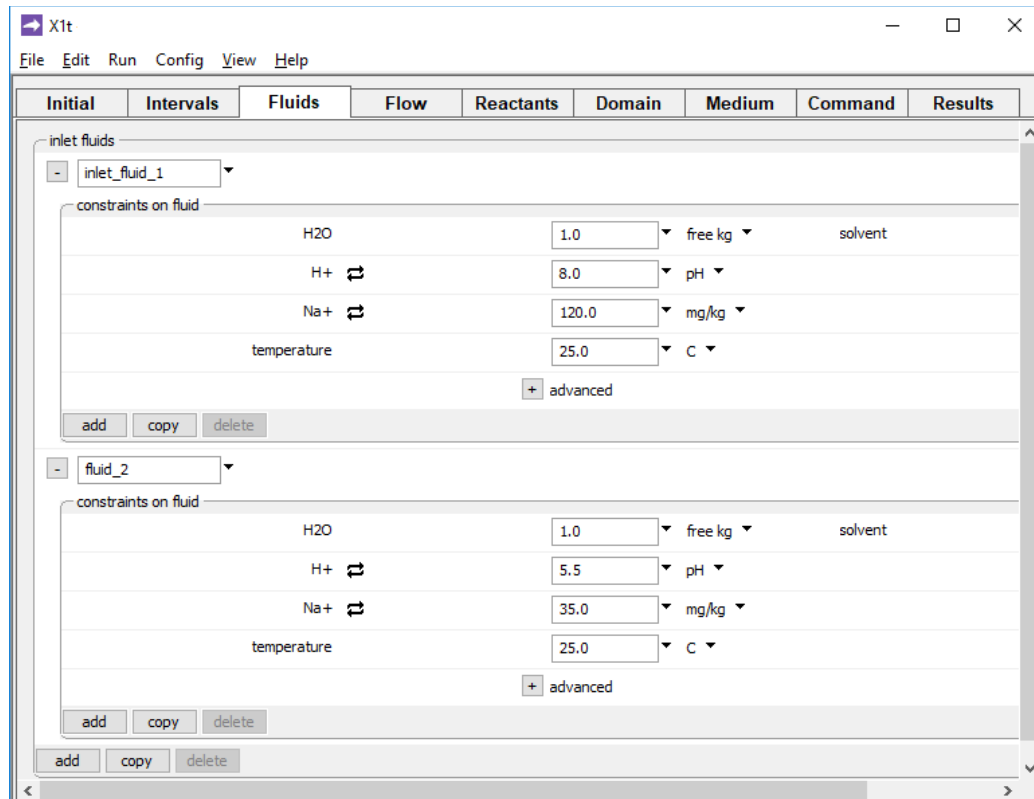
In this way, you do not need to match mineral mass to the size of nodal blocks in the simulation. If you double the number of nodal blocks, for example, the size of each block decreases by one half; you would prefer mineral mass within each block to likewise be cut in half. Such behavior is assured when you set mineral abundance in relative rather than absolute units.

2.8 Boundary fluids

In **X1t** and **X2t** simulations, unreacted fluid enters, and reacted fluid leaves the domain at its boundaries. In the case of **X2t**, fluid can furthermore enter and exit the domain at wells.

You set the composition and temperature of a boundary fluid—a fluid that enters the domain across its edge and through a well—on the **Fluids** pane. You can create as many boundary fluids as you like. A given boundary fluid may be carried for the duration of the run, or used only during a specific interval within the simulation; it might apply in many places, or only at a certain boundary or a single well.

To create a boundary fluid, move to the **Fluids** pane and click **add**, then click the **+** and set a name for the fluid. You can then set the fluid's composition by constraining each basis entry, much as you would in **SpecE8** or **React**. In the example below,



we've created two fluids, "inlet_fluid1" and "fluid_2", and constrained their compositions.

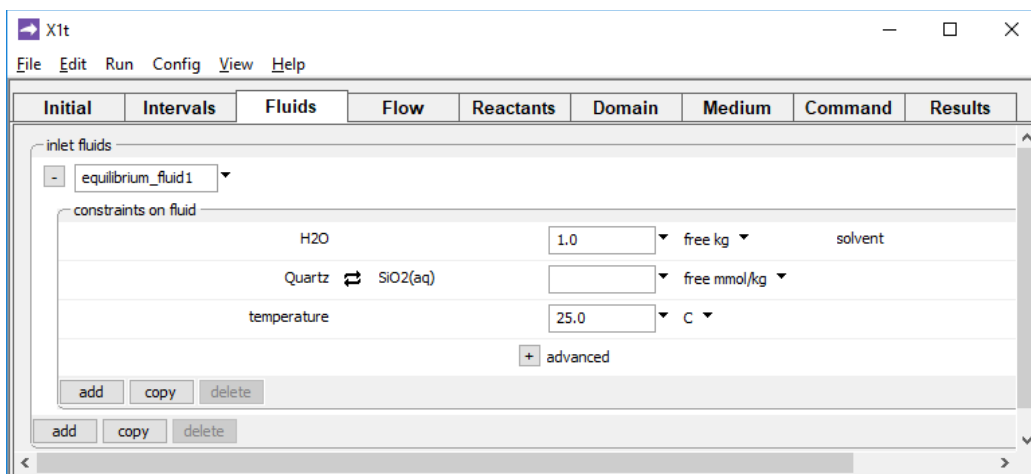
To use the command line, enter a `scope` command followed by a name for the fluid, then constrain each basis entry. The commands

```
scope inlet_fluid1
  pH = 8
  Na+ = 120 mg/kg
(and so on)
scope injection_fluid9
  pH = 5.5
  Na+ = 35 mg/kg
(and so on)
scope
```

16 *GWB Reactive Transport Modeling*

create the two fluids shown above.

To set a boundary fluid to equilibrium with a mineral, you swap the mineral into the basis



as you would when configuring **SpecE8** or **React**. In such cases, it is not necessary to set the amount of the mineral present, since the amount of a mineral in equilibrium with a fluid does not affect the fluid's composition. The following commands


```
scope equilibrium_fluid1
  swap Quartz for SiO2(aq)
scope
```

configure the same boundary fluid.

You set a polythermal simulation in the same fashion:

```
T = 25 C
scope fluid1
  T = 60 C
scope fluid2
  T = 100 C
scope
```

In this case, boundary fluids “fluid1” and “fluid2” are set to 60°C and 100°C, respectively, whereas the temperature of the initial system and any other boundary fluid is 25°C.

You can use the  button on the various basis panes, or the `scope` command, to transfer the basis constraints from one aspect of the calculation to another. For example, the command

```
scope boundary1 = initial
```

sets the composition and temperature of the boundary fluid “boundary1” to be the same as the fluid in the initial system, and the command

```
scope boundary2 = boundary1
```

sets the second boundary fluid to be the same as the first.

2.9 Boundary conditions

The programs, by default, use “inlet/free outlet” boundary conditions. At an inlet, which is wherever fluid crosses a boundary into the domain, the composition is set to that of the boundary fluid. Fluid crossing the boundary carries mass into the domain by advection; hydrodynamic dispersion and molecular diffusion furthermore add and remove mass from the domain, in response to concentration gradients between the boundary fluid and fluid within the domain.

At an outlet, which is a point where fluid exits the domain, fluid composition, by default, is fixed at the composition of the fluid within the nodes along the boundary. Chemical mass here is advected from the domain, but there is neither a dispersive nor diffusive flux. Conceptually, it is as if fluid flowing from the domain bathes the outside of the boundary.

The codes treat a boundary where discharge is zero as a free outlet. Fluid, by definition, does not cross such a boundary, and diffusion and dispersion are precluded at an outlet, so mass is, by default, not transported across a no-flow boundary.

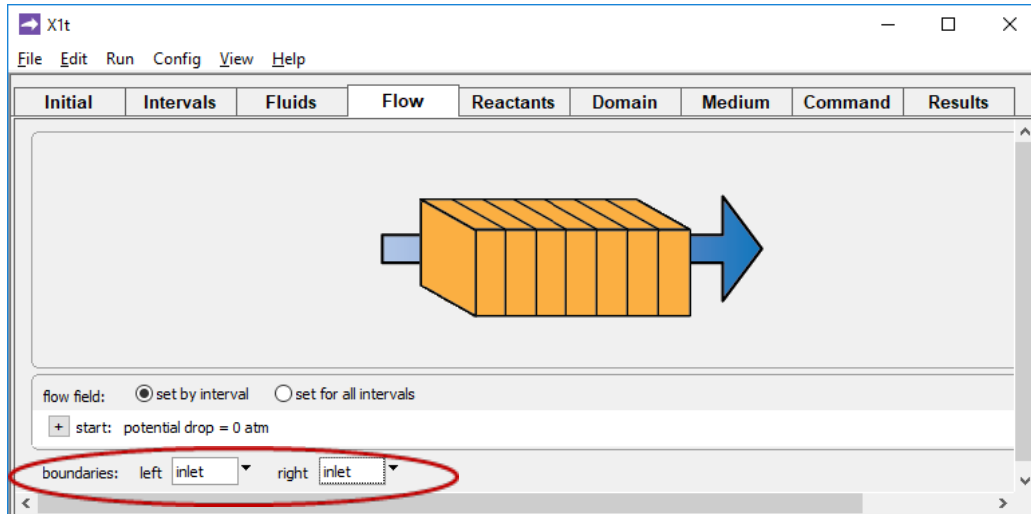
In polythermal runs, heat transfer across the boundaries behaves in a similar fashion. At inlets, temperature along the boundary is held constant at the temperature of the boundary fluid, so that heat energy crosses the boundary by conduction as well as advection. There is, however, no conduction at outlets, because there is no temperature gradient.

At injection wells, in **X2t**, a boundary fluid of prescribed composition enters the domain. Production wells, in contrast, extract fluid from where the well was completed within the domain, at a composition corresponding to the current point in the simulation progress.

You can modify the programs’ default behavior so that a boundary behaves as an inlet, open to diffusion and dispersion, or as an outlet lacking such transport, regardless of the

18 *GWB Reactive Transport Modeling*

direction the fluid flows. To do so, you use the “left” and “right” conditions on the **Flow** pane:



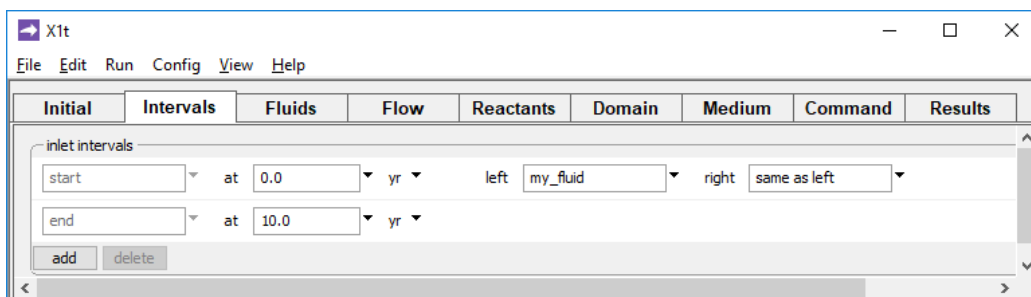
Here, both sides of the domain are set to behave as inlets, open to not only zero-order, but also first-order transport. The commands

```
left = inlet  
right = inlet
```

serve the equivalent purpose.

2.10 Reaction intervals

X1t and **X2t** simulations invariably begin with a reaction interval labeled “start”, and the termination point is called “end”. In the simplest case, a simulation consists of a single reaction interval. In this case, you go to the **Intervals** pane



to set the beginning and ending times of the simulation, and to select the fluids to appear at the domain's boundaries.

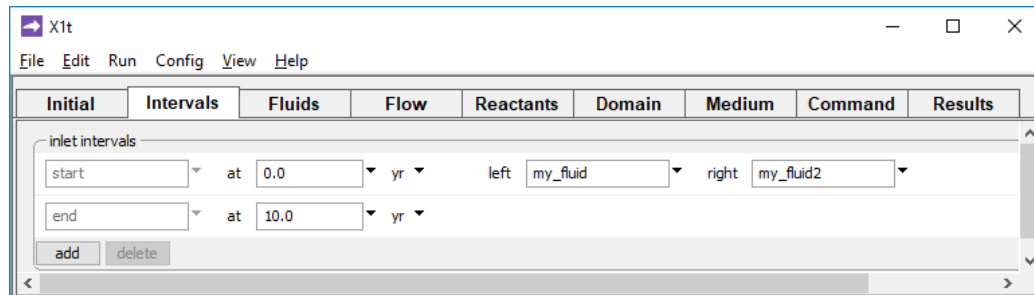
You can also set reaction intervals from the **Command** pane:

```
time start 0 years, end 10 years
interval start fluid = my_fluid
```

or, equivalently

```
interval start 0 years, fluid = my_fluid
interval end 10 years
```

You may choose to set distinct fluids



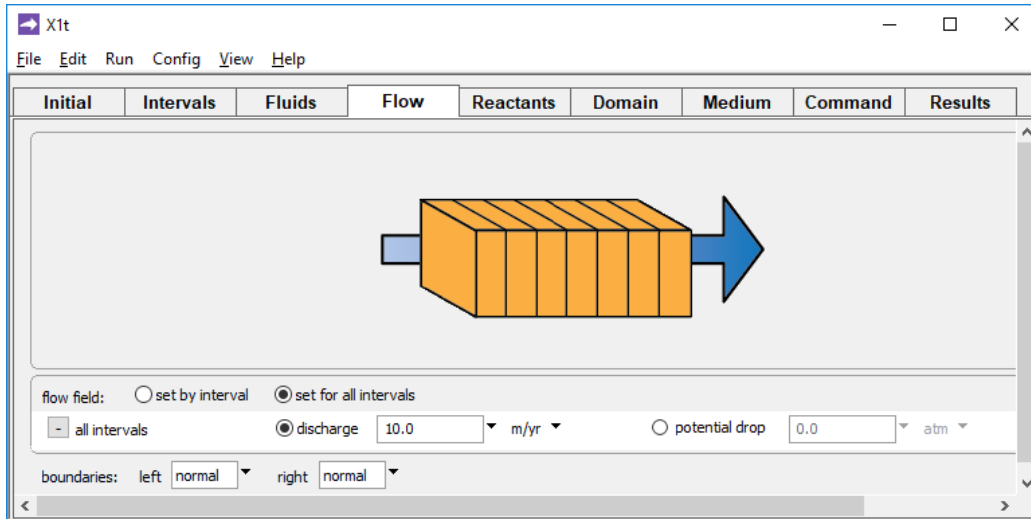
at the domain's left and right. The commands

```
interval start 0 years, left fluid = my_fluid1 right \
    fluid = my_fluid2
```

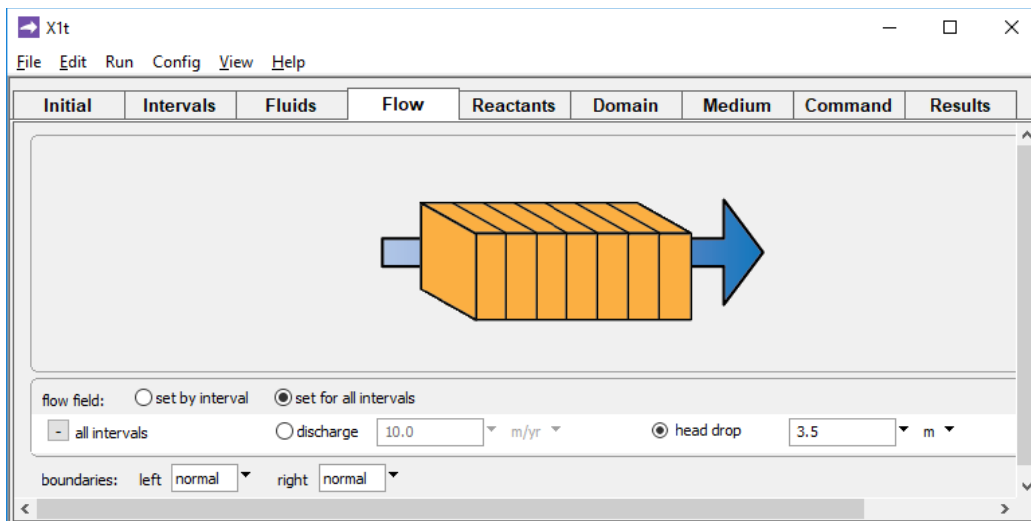
set up the same reaction interval.

20 *GWB Reactive Transport Modeling*

You set the flow rate on the **Flow** pane, or using a command. You can specify flow either directly in terms of specific discharge,

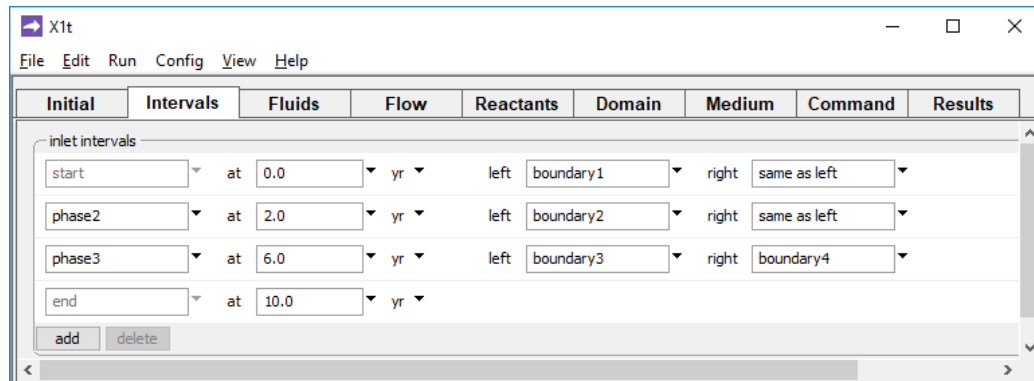


or indirectly in terms of the hydraulic head or potential drop across the domain.



In the latter case, the program calculates the flow field from the head drop, the permeability field, and the fluid's viscosity.

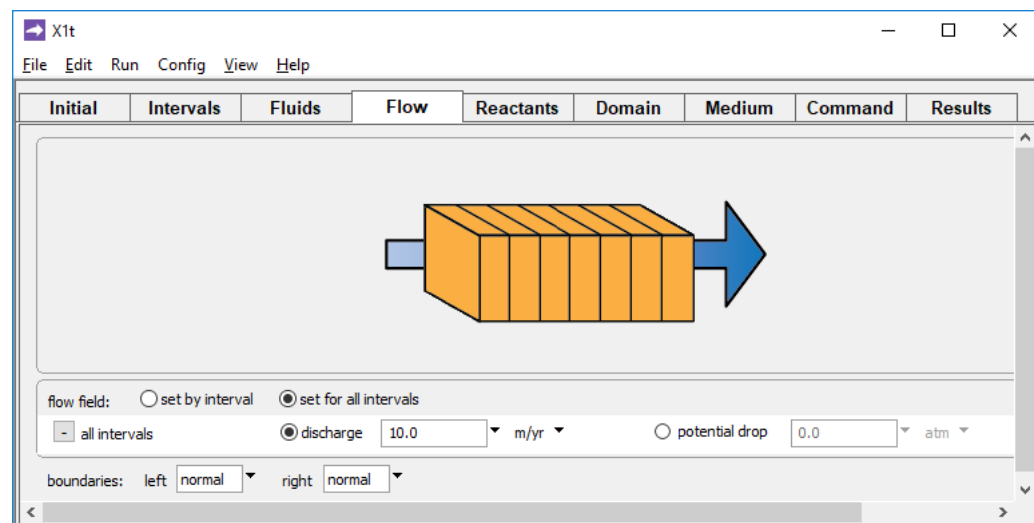
A simulation, however, can consist of any number of reaction intervals, each of which is defined by a given starting time, boundary fluid, and flow regime. For example,



sets three reaction intervals—“start”, “phase2”, and “phase3”—beginning at 0 years, 2 years, and 6 years, during which the fluids “boundary1”, “boundary2”, “boundary3”, and “boundary4” lie along the domain boundaries; the simulation in the example continues to 10 years.

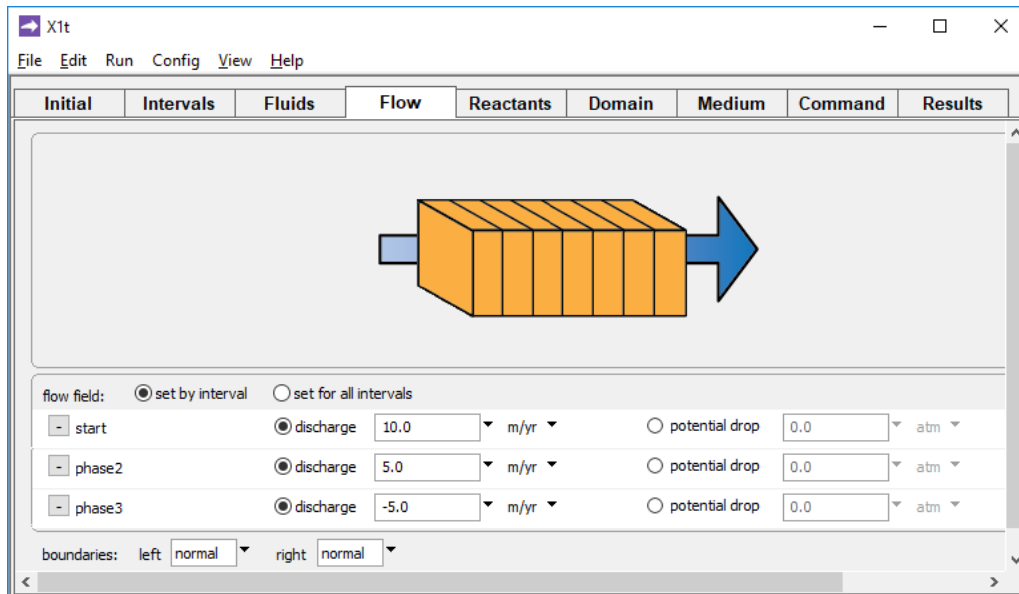
```
interval start 0 years, fluid = boundary1
interval phase2 2 years, fluid = boundary2
interval phase3 6 years, left fluid = boundary3 \
    right fluid = boundary4
interval end 10 years
```

Continuing, you can set discharge globally, for all the intervals



or by individual interval.

22 GWB Reactive Transport Modeling



In the latter example, flow slows, then changes direction as the simulation proceeds.

Finally, you may choose to prescribe the boundary fluid with a `discharge` command

```
discharge phase2 = 5 m/yr, fluid = boundary2
```

rather than an `interval` command.

2.11 Kinetic reactions and gas buffers

You set the kinetic reactions, fugacity buffers, and so on that operate within the domain over the course of the simulation on the **Reactants** pane. Here, you can define

- simple reactants that are added to the fluid at a set rate over the course of the simulation
- activity and fugacity buffers, such as fixing CO₂ fugacity within the domain
- sliding buffers, such as varying the CO₂ fugacity within the domain over the course of the simulation
- kinetic reactions by which minerals dissolve and precipitate
- kinetic reactions by which aqueous and surface complex species dissociate and associate
- the kinetics of gas transfer
- kinetic redox reactions
- the kinetics of microbial metabolism and growth

In each case, you define reactants much the same way you would in a **React** simulation.

You can set the reaction rate for simple reactants in units of mass or volume added per unit time per unit volume of the domain. For example, you could specify that a mineral dissolve at a rate expressed in cm^3 mineral per cm^3 of the porous medium per year.

You can set many values as field variables. These values include:

- the amount of a reactant or kinetic mineral present
- the cutoff value for a simple reactant
- the target activity or fugacity for a sliding buffer

From the command line, you set reactants using the `react` and `kinetic` commands, as you would in **React**.

As a final note, there is a danger in reactive transport modeling of setting overly rapid rates for kinetic reactions. If, in a simulation, a kinetic reaction proceeds very quickly relative to the simulation's time span, it will force time steps so small that the simulation never reaches completion, or it will make the simulation unstable. You should avoid specifying rapid kinetic rate laws. Instead, set such reactions as equilibrium buffers, since they will quickly approach equilibrium in the simulation.

2.12 Porosity evolution

In a simulation in which minerals in the domain precipitate and dissolve, the sediment porosity evolves to reflect the net change in mineral volume. Depending on how you have configured the simulation, the evolving porosity may affect the calculated permeability and, hence, flow rates and patterns within the domain.

If you do not set the porosity explicitly on the **Medium** pane, or with the `porosity` command, the program figures the porosity at each node from the volumes of fluid and “rock”, the latter being the sum of the volumes of the minerals in the nodal block. The minerals include both the equilibrium minerals and solid solution end members swapped into the basis and the reactant minerals and end members—those reacting according to kinetic rate laws and any set as simple reactants.

For simulations in which you set an explicit value for porosity, the program, when setting the initial conditions, figures the difference at each node between the rock volume corresponding to the porosity value and the volume of minerals in the block. This volume is then carried through the simulation as “inert” or nonreactive volume.

In either case, the program tracks how porosity evolves over the course of the simulation on the basis of the change in mineral volume. Where the sum of the volumes of the equilibrium and reactant minerals increases, the porosity decreases, and where mineral volume decreases, porosity increases.

A handy use of the inert volume feature is simulating reaction in an unsaturated medium. The unsaturated fraction of the medium is defined by setting porosity to a value less than the volume fraction remaining after accounting for the mineral volume. The

program figures the inert volume, which can be thought of as the volume of the gas phase, and carries it through the simulation.

For example, in the case

```
porosity = 25%
5 volume% K-feldspar
5 volume% Albite
50 volume% Quartz
```

15% of the porous medium can be thought of as being occupied by a gas phase. Then, the fugacities of the reactive gases in the gas phase can be set as fixed or sliding buffers:

```
fix f CO2(g)
fix f O2(g)
```

In this example, the CO₂ and O₂ fugacities are held to their initial values, as calculated when the initial conditions are determined.

2.13 Permeability correlation

The models use an empirical correlation to calculate the permeability of the porous medium from the porosity and, optionally, the mineral content of the medium. When **X1t** and **X2t** determine the flow field from the boundary conditions you specify, the program calculates the pattern of discharge across the domain at each step in the simulation, using the permeability and fluid viscosity computed at each node. In this case, the evolution of the permeability has a direct effect on the simulation results.

When, in **X1t**, you prescribe discharge directly, or, in **X2t**, you import the flow field instead of calculating it, the programs compute values for permeability at each node over the course of the simulation, but the values have no effect on the simulation results.

There is no general relationship by which the permeability of actual sediments or rocks varies with porosity and mineralogic composition. For a specific suite of sediments or rocks, however, it is commonly possible to establish a statistical correlation among these variables. In constructing a model, it is important to remember that all such correlations are empirical, not functional constraints.

The programs use a correlation of the form

$$\log k_x = A\phi + B + \sum_m A_m X_m \quad (2.4)$$

to calculate permeability. Here, k_x is permeability along x in darcys ($1 \text{ darcy} \approx 10^{-8} \text{ cm}^2 = 1 \mu\text{m}^2$), ϕ is porosity (expressed as a volume fraction), A , B , and A_m are empirical constants, and X_m is the volume fraction of an arbitrary set of minerals indexed by m .

By default, the values for A and B in the correlation are set to 15 and -5 , respectively, and no minerals are carried. The default correlation, then, is

$$\log k_x(\text{darcy}) = 15\phi - 5 \quad (2.5)$$

which describes a trend that has been observed in sandstone. The default settings, of course, are of no general significance.

You set the permeability correlation on the **Medium** pane, entering A and B in the boxes labeled “A (porosity)” and “B (intercept)”. To add a mineral to the correlation, click on **add**, select the desired mineral from the pulldown list, and set the corresponding coefficient. To delete a mineral, select the entry in question by clicking on it, and press **delete**.

For example, you would set a hypothetical correlation

$$\log k_x = 12\phi - 3 - 40 X_{\text{kaolinite}} \quad (2.6)$$

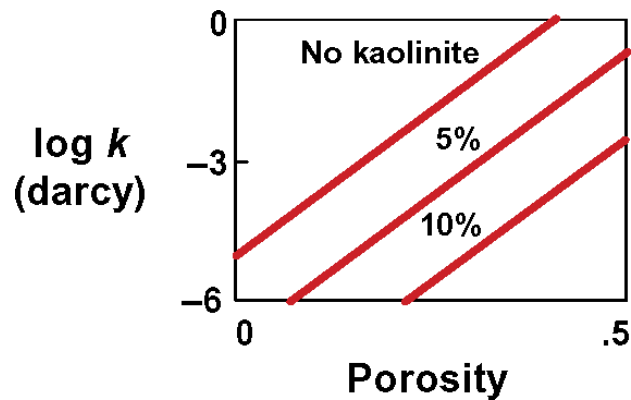
as shown below.

permeability: $\log k_x (\text{darcy}) = (A \times \text{porosity (volume fraction)}) + B + (A1 \times \text{Kaolinite (vol. frac.)})$

A (porosity)	+	12.0	▼
B (intercept)	+	-3.0	▼ darcy ▼
A1 (Kaolinite)	+	-40.0	▼

add **delete**

By this correlation, permeability increases with increasing porosity, but the presence of kaolinite serves to decrease the permeability, as shown below:



Alternatively, you can set the permeability correlation with the `permeability` command and arguments `porosity` (to set the value of A), `intercept` (to set B), and the names of the minerals k to be included in the correlation (to set A_m).

```
permeability porosity = 12, intercept = -3, Kaolinite = -40
```

To remove a term from the correlation, set its value to `?`. For example, the command

```
permeability Kaolinite = ?
```

removes the term for kaolinite from the correlation.

When considering solid solutions, the contribution of end members and tranches to the permeability follows from settings for the corresponding minerals. You can also include discrete tranches directly in the correlation.

2.14 Groundwater flow

In **X1t**, you define the rate of groundwater flow through the domain by setting either the fluid discharge into the medium, or the driving force for flow—the drop in head or hydraulic potential across the domain.

If you set the discharge directly, a positive value is taken as the flow rate into the left side of the domain, and a negative value gives flow into the right side. If you set the driving force, the model calculates discharge at each step in the simulation. Discharge, in this case, varies depending on the evolving values calculated for permeability and fluid viscosity.

In **X2t**, you most commonly let the program calculate the groundwater flow field as it evolves over the course of the simulation. You set the left and right boundaries to be either open or closed to flow, or to be crossed by a certain discharge. The top and bottom

boundaries are closed to flow. You then set the head or potential drop across the domain and either the production or injection rate, or the head at each well.

You can also import the flow field from another program. In this case, you set up a table of the fluid discharge from node to node along x , and another for flow along y . **X2t** reads the tables and uses the values they contain to set the flow field. In this case, fluid may enter or leave the domain across any of the boundaries, not just the left and right sides.

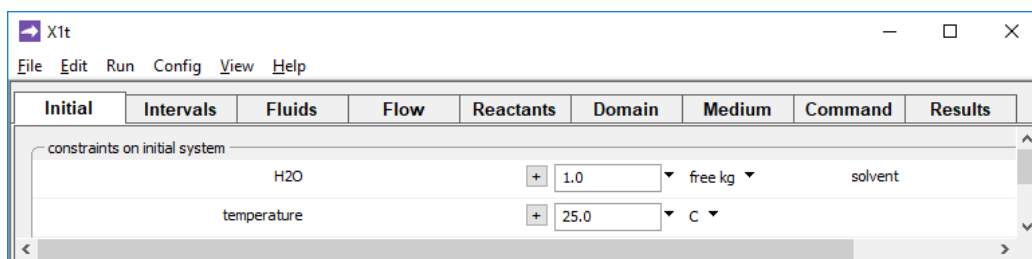
You can, as a third alternative, import the flow field from the results of a MODFLOW simulation. MODFLOW is a popular program for modeling groundwater flow. For details, see the [Importing From MODFLOW](#) appendix to this guide.

For details on setting the flow field in an Xt simulation, see the [Using X1t](#) and [Using X2t](#) chapters of this guide.

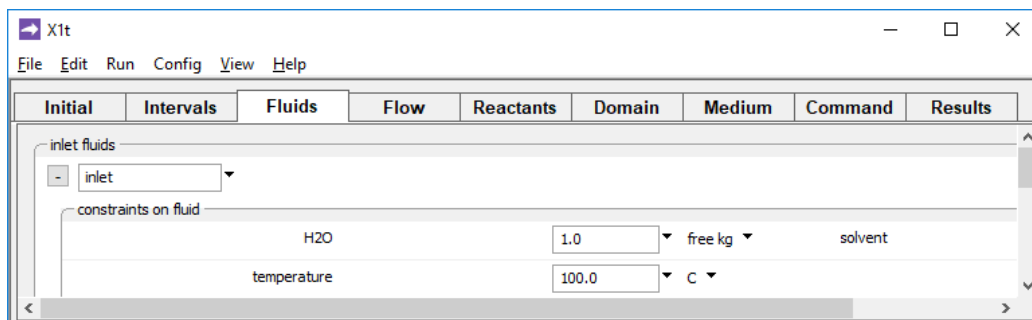
2.15 Polythermal simulations

You set the initial temperature of the domain in C on the **Initial** pane by entering a value next to the “temperature” label. To set polythermal initial conditions, click on the **+** next to this label and define temperature as a field variable. You set the temperatures of the boundary fluids in your simulation on the **Fluids** pane.

For example, the settings on the **Initial** pane



and on the **Fluids** pane



specify that a 100°C boundary fluid named “inlet” invades a domain with an initial temperature of 25°C.

28 *GWB Reactive Transport Modeling*

You can also set values for these temperatures from the command line, using the `temperature` (or `T`) command. The command

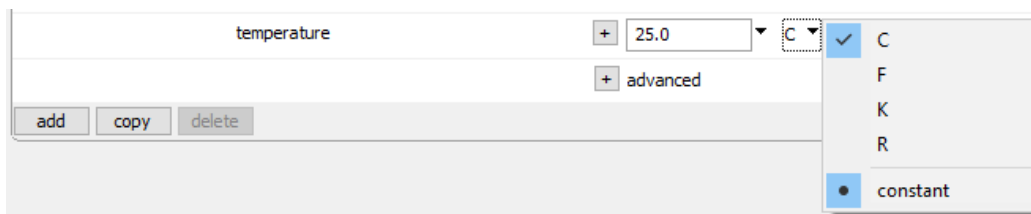
```
temperature initial = 25 C, inlet = 100 C
```

specifies the same conditions as mentioned above. Alternatively, the commands

```
scope initial
  temperature = 25 C
scope inlet
  temperature = 100 C
scope
```

serve the same purpose.

X1t and **X2t** can trace a second type of polythermal simulation in which temperature varies across the domain but remains invariant over the course of the run. This type of simulation is intended to represent the case in which the domain controls the temperature of the migrating fluid. In a simulation of flow through a fracture, for example, heat conduction normal to the direction of flow might hold temperature constant. To trace a simulation of this type, select the “constant temperature” option on the **Initial** pane



or issue the command

```
temperature constant
```

The command

```
temperature constant = off
```

deselects the option.

You can alter several values used in simulating heat transfer on the **Medium** pane. Variables “cpw” and “cpr” are the heat capacities of water and rock grains in cal/g °C. You

can set thermal conductivity in cal/cm/s °C as a field variable. These values can also be set from the command line, with the `cpw`, `cpr`, and `thermal_cond` commands.

2.16 Internal heat production

X1t and **X2t** can trace polythermal simulations in which heat is produced or consumed internally in the domain, for example by exothermic or endothermic chemical reaction, or by radioactive decay. You specify the rate of heat production per unit volume of the domain (in, for example, J/m³/s) in your input as a field variable, and the program at each time step accounts for the effect of production on the distribution of temperature within the domain.

You set the production rate on the **Medium** pane, or with the `heat_source` command. The input

internal heat source

temperature: minimum °C maximum °C

+ J/m³/yr ▼

specifies that heat is produced over the course of the simulation at a constant rate. A negative value for the production rate indicates that heat is consumed rather than produced. Alternatively, the command

```
heat_source = 200 J/m3/yr
```

does the same thing.

In a polythermal simulation without an internal source of heat, temperature everywhere falls within the temperature range set in your input data. If, for example, you set initial temperature in the domain to 25°C and the temperature of the inlet fluid to 100°C, temperature over the course of the simulation will fall in the range 25°C–100°C.

In a simulation accounting for heat production, in contrast, the program does not know in advance the range of possible temperatures that might be encountered in the run. The program, therefore, requires that temperature limits be specified or assumed at the start of the run.

The settings for the temperature limits have two practical consequences. If temperature at any nodal block falls more than 5°C outside the range specified, the program will issue an error message and discontinue the simulation. More significantly, the program will load only those chemical species (aqueous species, minerals, and so on) for which thermodynamic data spanning the temperature range is available in the thermo dataset, unless the “extrap” option is set. Therefore, how you set the allowed temperature range can affect which chemical species are included in a simulation.

By default, **X1t** and **X2t** assume the temperature range specified in the header blocks of the thermodynamic dataset; this range is 0°C–300°C in “thermo.tdat”, the default thermo dataset. You can control the allowed temperature range explicitly on the **Medium** pane, or with the `temp_min` and `temp_max` keywords in the `heat_source` command. For example, the following input

internal heat source

temperature: minimum °C maximum °C

J/m3/yr

sets an allowed temperature range of 25°C–200°C. The command

```
heat_source = 200 J/m3/yr, temp_min = 25 C, temp_max = 200 C
```

does the same.

2.17 Dual porosity model

Programs **X1t** and **X2t** include a dual porosity feature in which each nodal block in the simulation is divided between a free-flowing zone and a collection of stagnant zones. Fluid and heat in the simulation move from nodal block to nodal block only within the free-flowing zone. Within a nodal block, however, water and solute can diffuse into and out of the stagnant zones. In a polythermal run, heat can be conducted from the free-flowing zone to the stagnant zones, and vice-versa.

You configure a dual porosity model on the **Config → Dual Porosity...** dialog, or with the `dual_porosity` command. You assign the geometric configuration and characteristic dimensions of the stagnant zones, and, if you wish, their physical properties. The stagnant zones occupy a fraction χ_{stag} (keyword `volfrac`) of a nodal block’s bulk volume, so the free-flowing zone takes up fraction $1 - \chi_{stag}$.

In a dual porosity model, it’s important to remember that values such as porosity ϕ , permeability k , and groundwater velocity v refer to conditions within the free-flowing zone, not the domain as a whole. Groundwater discharge q , in contrast, reflects the rate fluid flows across a unit area of the entire domain, including the stagnant as well as free-flowing zones. Specific discharge, then, can be calculated from Darcy’s law written

$$q = -(1 - \chi_{stag}) \frac{k}{\mu} \left(\frac{\Delta\Phi}{\Delta x} \right) \quad (2.7)$$

and groundwater velocity is given as

$$v = \frac{q}{\phi(1 - X_{stag})} \quad (2.8)$$

since flow occurs only within the free-flowing zone.

The stagnant zones are deployed in one of three geometries. Choosing the spheres geometry (keyword `spheres`, the default), the stagnant fraction is composed of spheres of radius δ (`radius`). For the block geometry (`blocks`), the fraction is made up of cubic blocks of half-width δ (`half-width`). And in the fractures geometry (`fractures`), flow and transport occur only through discrete fractures along the x direction. In this case, the stagnant fraction is composed of slabs of half-width δ (`half-width`), separated by the fractures. For **X2t** simulations, the latter choice precludes flow and transport along y , since the free-flowing zone is composed of fractures oriented along x .

The most important quantity in a dual porosity model is the cumulative surface area of the stagnant zones, which is the contact area with the free-flowing zone; geometric details, such as the exact topology of the stagnant zones, have little effect on the simulation results. For spheres and blocks, the surface area A_s per unit volume of the domain is given by $3X_{stag}/\delta$; for fractures, it is X_{stag}/δ . The best strategy for constructing a meaningful dual porosity model, then, is to choose values for X_{stag} and δ that give the desired contact area.

As an example, the following input to the **Config → Dual Porosity...** dialog

The screenshot shows the 'Dual porosity' dialog box with the following settings:

- ☒ dual porosity
- geometry: ☒ spheres, ☐ blocks, ☐ fractures
- volume fraction: 0.6
- nsubnode: 5
- radius: 0.5 m
- diffusion length: 1 m
- porosity: 0
- retardation factor: 1
- diffusion coefficient: 1 cm²/s
- thermal conductivity: 1 cal/cm/s/C
- theta: 0

Buttons at the bottom: OK, Apply, Cancel, Reset.

sets a dual porosity model in which the stagnant zones are composed of spheres 1 m across; the spheres take up 60% of the domain's volume. You must, at a minimum, set values for X_{stag} and δ ; as field variables, these values can vary across the domain.

32 GWB Reactive Transport Modeling

The command

```
dual_porosity spheres, half-width = 1/2 m, volfrac = 60%
```

configures the same thing.

The programs work by solving for the diffusive profile within the stagnant zones, using finite differences. The profile extends inward from the contact with the free-flowing zone along a spherical coordinate when the geometry is set to spheres, and along a linear coordinate for the blocks and fractures geometries. In a polythermal simulation, the programs similarly solve numerically for the conductive temperature profile. To accommodate numerical solution, the programs divide the stagnant zones within each nodal block into equally spaced subnodes. The software uses 5 subnodes by default, but you can set this number N_{stag} to another value with the `Nsubnode` keyword. Setting too large a value for N_{stag} can slow the simulation considerably, by requiring extra computing effort at each nodal block, and forcing small time steps, as described below.

In many applications, the simulation's time span is short relative to the time needed for the solute to diffuse more than a short distance. In such cases, solute may diffuse into or out of only the stagnant zones' outermost skin. To avoid wasting computing time and to derive a more accurate solution, you should set the model to consider diffusion (and, in polythermal runs, heat conduction) in only the outermost part of the zones, over a diffusion length δ_{diff} , using keyword `diff_length`. An appropriate value for this variable can be estimated from the simulation's duration t as $\delta_{\text{diff}} \gtrsim 2\sqrt{D^*t}$, or $\delta_{\text{diff}} \gtrsim 2\sqrt{K_T t}$ for heat conduction.

The input

Dual porosity

☒ dual porosity

geometry ☒ spheres ☐ blocks ☐ fractures

volume fraction

nsubnode

radius m

diffusion length cm

porosity

retardation factor

diffusion coefficient cm²/s

thermal conductivity cal/cm/s/C

theta

OK Apply Cancel Reset

sets the program to consider mass and heat exchange within the outermost 1 cm of the stagnant zones, dividing this length into 10 subnodes. The command

```
dual_porosity Nsubnodes = 10, diff_length = 1 cm
```

does the same thing.

You can further specify the diffusion coefficient D^* (keyword `diff_coef`, in cm^2/s), porosity ϕ (`porosity`), and retardation factor R_f (`retardation`) within the stagnant zones using the dialog

The screenshot shows a 'Dual porosity' dialog box with the following settings:

- ☒ dual porosity
- geometry: ☒ spheres, ☐ blocks, ☐ fractures
- volume fraction: 0.6
- nsubnode: 5
- radius: 0.5 m
- diffusion length: 1 cm
- porosity: (empty)
- retardation factor: 2.0
- diffusion coefficient: 1.0e-6 cm^2/s
- thermal conductivity: (empty) $\text{cal}/\text{cm}/\text{s}/\text{C}$
- theta: (empty)

Buttons: OK, Apply, Cancel, Reset

or with the command

```
dual_porosity diff_coef = 10^-6, retardation = 2.0
```

These values, which as field variables can vary with position in the simulation, control how quickly solute and solvent diffuse into and out of the stagnant zones.

In a polythermal simulation, you can likewise set thermal conductivity K_f in $\text{cal}/\text{cm}/\text{s}/\text{C}$ with keyword `thermal_con`; the conductivity controls the rate of heat exchange. If you do not set a value for D^* , ϕ , or K_f , the program uses the current setting from the free-flowing zone. If not specified, R_f is taken as 1, reflecting an absence of retardation.

You can examine the diffusive profiles within the stagnant zones by including the profiles in the print-format output dataset. Go to **Config → Output...** and fill the “stagnant zone” checkbox, then click **Apply**.

The command

```
print stagnant = long
```

similarly enables such output.

In solving for diffusion within the stagnant zones, the numerical procedure is weighted in time according to a variable θ (“theta”). Setting $\theta = 0$ invokes the “explicit method” in which the mass flux into and out of a stagnant zone is computed at the onset of each time step. A time step can be completed quickly by this method, but the step sizes are limited by the stability criterion

$$\Delta t \leq \frac{1}{2} \left(\frac{\delta_{\text{diff}}}{N_{\text{stag}}} \right)^2 \frac{R_F}{D^*} \quad (2.9)$$

Note that by this relation, increasing the value of N_{stag} quickly reduces the limiting step size.

Specifying $\theta \geq 1/2$ assigns an unconditionally stable “implicit method” in which the program also considers mass fluxes at the end of the time step. This method requires more computing to complete a step, but, since it is not subject to a stability criterion, can be faster overall if fewer steps are needed to trace a simulation.

The programs, by default, select a value for θ automatically, using $\theta = 0$, unless the explicit method would force too many more time steps than would otherwise be needed. In that case, **X1t** and **X2t** set an implicit solution, with $\theta = 0.6$. You can override this behavior by setting a value for “theta” directly. You might, for example, set $\theta \geq 0.5$ in order to maintain long time steps, to minimize numerical dispersion.


2.18 Mobile colloids

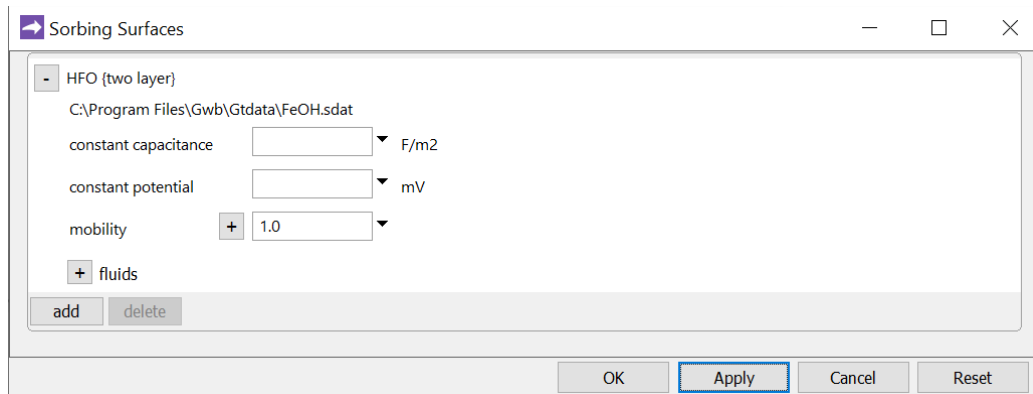
Complexing mineral surfaces in **X1t** or **X2t** simulations can be allowed to move through the domain along with the fluid as it advects and disperses. In this case, the surfaces become mobile colloids. A mobile colloid, specifically, consists of the mineral or minerals associated with a surface that moves with the fluid, as well as the ions that have complexed with the surface.

By their nature, mobile colloids are limited to complexing surfaces, which are defined by surface datasets of the two-layer, triple-layer, and CD-MUSIC types, as opposed to sorbing surfaces. Sorbing surfaces—surfaces of the K_d , Freundlich, Langmuir, and ion exchange types—are not associated with specific minerals and, hence, cannot be used to construct a mobile colloid.

Each complexing surface in a simulation has a mobility, which, by default, is 0. Mobility refers to the fraction of the surface in question that can in the model move by advection and dispersion. A mobility of 0 sets the surface to be stationary, whereas a surface with a mobility of 1 moves freely. A surface can have a value intermediate between 0 and 1. Intermediate values might arise when a portion of the surface has aggregated or is attached to the medium, for example, or when colloidal motion is impeded by electrostatic interactions.

To set a mobile colloid, you first read in a surface complexation dataset using the **File → Open → Sorbing Surfaces...** (or **Config → Sorbing Surfaces...**) dialog, or using the `surface_data` command. Only datasets with model type “two-layer”, “three-layer”, or “cd-music” in the header are surface complexation models and can be used to form a mobile colloid. You then set a value in the mobility field. Alternatively, you can use the `mobility` command, referencing the label of the surface in question, to set the colloid’s mobility. If you omit the label, the program will assume you are referring to the surface complexation dataset most recently added to the simulation.

To add the Dzombak and Morel dataset for ion complexation with the hydrous ferric oxide surface, for example, and then set the surface to be mobile, go to **File → Open → Sorbing Surfaces...** to launch the dialog, then click  and browse for the file “FeOH.sdat”. Set the mobility to “1”, then click **Apply**.

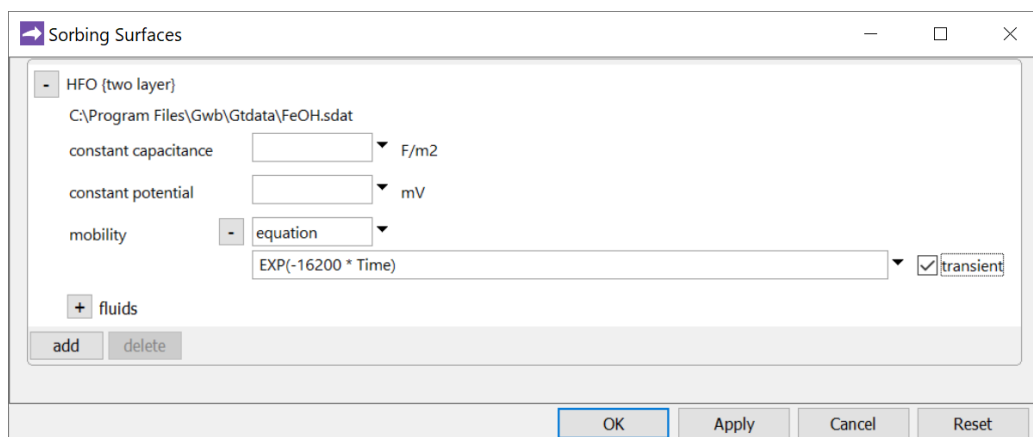


Alternatively, use the commands

```
surface_data FeOH.sdat
mobility HFO = 100%
```

HFO here is the label for the ferric oxide surface, as specified in the header of “FeOH.sdat”.

Mobility is cast in **X1t** and **X2t** as a transient field variable, which means you can have the software calculate mobility at each nodal block according to an equation, script, or compiled function you provide. When you set the `transient` keyword, the program, upon undertaking each time step in the simulation, evaluates mobility across the domain. In contrast, in the default steady behavior, the program evaluates mobility at each block just once, at the start of the run. As an example, click the **+** button next to “mobility”, select “equation” from the pulldown, type in the expression below, check ☒ the transient box, then click **Apply**.




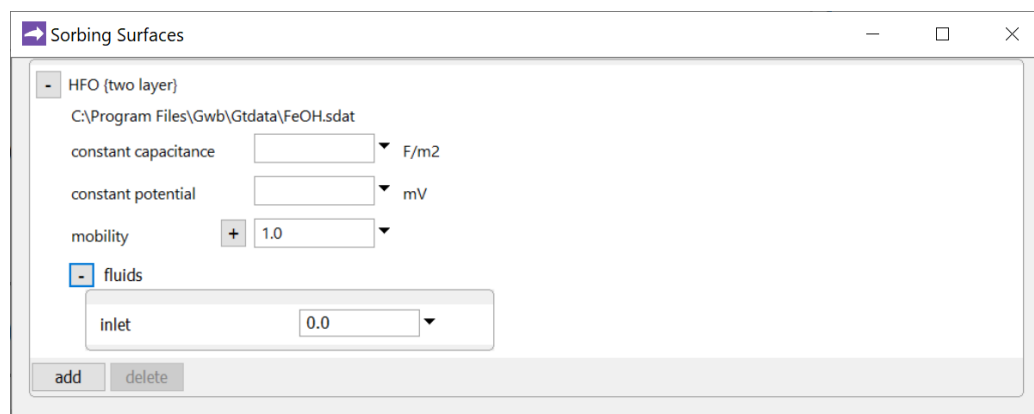
to set colloid mobility to decay exponentially with the passage of time.

The command

```
mobility HFO = eqn "EXP(-16200 * Time)" transient
```

is equivalent.

You can also define colloid mobility in boundary fluids. To do so, click the  next to “fluids” and set a value for one or more boundary fluids. The following settings



allow a colloid to move within the domain, but not be carried in the inlet stream. The sequence of commands

```
scope initial
  mobility HFO = 100%
scope inlet
  mobility HFO = 0%
scope
```

or more concisely

```
mobility HFO = 100%, inlet = 0%
```

is equivalent to the above setting. If you do not set a value, mobility in a boundary fluid is taken as the initial value assigned to the first interior node in the simulation.

The mobility of end members and tranches of solid solutions follow from the corresponding minerals. Setting one mineral contained in a solid solution mobile carries over to the paired mineral, as well as any tranches of the solution. If both end-member minerals sorb, their surfaces must have the same mobility set, whether the solid solution is continuous or discrete; the apps do not allow conflicting mobilities.



2.19 Stable isotope transport

Programs **X1t** and **X2t** can model in one and two dimensions, respectively, the transport of stable isotopes in reacting systems, accounting for the effects of advection, hydrodynamic dispersion, and molecular diffusion. The section **Fractionation of Stable Isotopes** in the *GWB Essentials Guide* describes how the GWB programs calculate the fractionation of stable isotopes, and a section of the same name in the *GWB Reaction Modeling Guide* shows how the programs model fractionation as a geochemical system reacts. You should familiarize yourself with those sections before continuing.

To make an isotope transport model, you set the bulk isotopic composition of the initial fluid, as well as the compositions of each reactant and segregated mineral, just as you would in **React**. In **X1t** and **X2t**, you further specify the isotopic composition for each boundary fluid included in the simulation.

Note that specifications must be inclusive: if you set $\delta^{18}\text{O}$ for the initial fluid, you need to set it as well for each boundary fluid, each oxygen-bearing reactant, and each segregated mineral that contains oxygen. Remember you may set isotopic compositions relative to any standard, as long as you are consistent, and the results are reported on that scale.

You configure a model of isotope transport from the **Config → Isotopes...** dialog, or on the **Command** pane. Suppose we want to set up a model with an initial fluid for which $\delta^{13}\text{C}$ is -5‰ on the PDB scale. The fluid maintains equilibrium with a CO_2 reservoir where $\delta^{13}\text{C}$ is $+2\text{‰}$, and with calcite, which has a composition of 0‰ , both also on the PDB scale. The calcite, furthermore, is segregated from isotope exchange. The $\delta^{13}\text{C}$ composition of an infiltrating boundary fluid, finally, is -22‰ .

In this case, start **X1t** and create a fluid “infiltrer” on the **Fluids** pane, and then fix the fugacity of $\text{CO}_2(\text{g})$ on the **Reactants** pane. Open the **Config → Isotopes...** dialog and expand the “initial” and “infiltrer” blocks by clicking on the  signs. Enter values within the “Carbon-13” blocks for the “initial”, “ $\text{CO}_2(\text{g})$ ”, and “infiltrer” fields. In the “segregated minerals” block, choose  → **Carbon-13...** → “Calcite”. Finally, set the carbon isotopic composition for “Calcite”. The dialog should look like

Click **OK** to complete the configuration. Alternatively, enter the commands

```
segregate Calcite
scope = initial
    carbon-13 initial = -5, CO2(g) = +2, Calcite = 0
scope = infiltrer
    carbon-13 initial = -22
```

on the **Command** pane.

When you run an isotope transport model, **X1t** and **X2t** begin by calculating the distribution of the isotopes in question at each node in the domain, in the same way that **React** sets the initial isotopic state. Over the course of the simulation, boundary fluids

enter the domain, displacing reacted fluid from it, and equilibrium and kinetic reactions proceed. In response, the distribution of the isotopes considered in the run changes over time.

The program writes the resulting isotopic compositions for species, minerals, and gases, as well the bulk compositions of the various phases, to a plot file “X1t_plot.xtp” or “X2t_plot.xtp”, where they are available for plotting by **Xtplot**. If you turn on the print option from the **Config → Output...** dialog, the program also writes the results in text format to a dataset “X1t_isotope.txt” or “X2t_isotope.txt”.

2.20 Time marching

Once **X1t** or **X2t** has successfully determined the boundary and the initial conditions and the groundwater flow field, it begins to march forward in time from the start of the simulation ($\xi = 0$) to the end ($\xi = 1$).

In taking a time step, the program automatically selects the time step length Δt according to a number of criteria:

- Numerical stability of the mass and heat transport equations, as described in the subsequent chapters of this guide, [Using X1t](#) and [Using X2t](#). The programs account for the maximum time step imposed by the rate of advection combined with the rates of dispersion, diffusion, and heat conduction.
- The intensity of any kinetic reactions you specify. In general, you should avoid setting overly rapid kinetic reactions, relative to the time span of the simulation, because they may result in very small time steps or numerical instability.
- Not stepping past specified points for writing results to the plot and print-format output files, the start of a new boundary interval, or the start or end of a well interval. The intervals in ξ for writing plot and print data are set by variables “dxplot” and “dxprint” on the **Config → Output...** dialog; starting times for boundary intervals are set on the **Intervals** pane, or with the `interval` command; and the time spans of well intervals are defined on the **Wells** pane, or with the `well` command.
- Any stability constraints imposed by the dual porosity feature, as described above, under [Dual porosity model](#).
- The initial step size, maximum step size, and maximum proportional increase in step size, as specified by variables “dx_init”, “delxi”, and “step_increase” on the **Config → Stepping...** dialog.

The programs can report which of the above factors limits each of the time steps taken over the course of a simulation. To enable this useful feature, go to **Config → Stepping...** and select the “explain steps” checkbox, or use the command

```
explain steps
```

To take full advantage of this feature, be sure to check the “Follow Output” option on the **Results** pane.

The computing time required to trace a simulation is approximately proportional to the number of nodes in the domain multiplied by the number of time steps taken. Numerical stability requires that the size of a time step vary with the physical dimensions of the nodal blocks. Setting a very fine numerical grid, therefore, can lead to lengthy simulations.

For example, if you increase the number of nodes in an **X1t** simulation from 100 to 1000, you have ten times as many nodal blocks, each one-tenth the length of the original block. It will take ten times as many steps to trace the simulation, and the computing time will increase one hundred-fold.

2.21 Multicore execution

Modern personal computers commonly contain more than one computing core. A computer with a single dual-core processor, for example, has two computing cores, and one with a pair of quad-core processors has eight. Most software executes as a single thread on one core, leaving the other cores to run other programs. A multithreaded program, however, gains a performance advantage by running on more than one core at the same time.

X1t and **X2t**, starting with GWB Release 8, are multithreaded applications, running in parallel on multicore computers. On a computer with eight cores, for example, invoking one of the programs spawns eight threads. The threads work on the calculations for different nodal blocks in parallel, speeding up the simulation considerably, relative to serial execution.

You can control the number of threads **X1t** and **X2t** spawn on the **Config → Stepping...** dialog, or with the `threads` command. The command

```
threads = 1
```

for example, sets the program to execute as a single-threaded application, whereas

```
threads = ?
```

returns the program to default behavior, spawning a thread for each computing core. Typing the command without an argument reveals the number of threads to be spawned and computing cores available.

If you set

```
pluses = multi
```

when the program finishes time stepping for a node, it issues a symbol distinct for each thread, instead of a “+” sign. You can set this output next to “progress” on the **Config → Stepping...** dialog. The default is “off” because printing the symbols for each node can slow down the simulation’s progress. The command

```
go single
```

runs the simulation on one processor, with a single thread.

After completing a simulation, the program reports the computing time spent executing parallel and serial sections of the code, as well as the clock time elapsed. Elapsed clock time, of course, depends on a number of factors, such as the computing load exerted by the operating system and various background tasks, as well as any other programs that may be running.

2.22 Running a model

Once you have configured a simulation, you are ready to run it. A number of example input files for **X1t** and **X2t** are installed with the GWB Professional package, in subdirectory “Script” within the installation directory (e.g., in “\Program Files\GWB\Script”). The example files have “.x1t” and “.x2t” extensions. To give one of the programs a test drive, double-click on one of the files.

You can initiate a model run in several ways: selecting **Run → Go**, pressing **Ctrl+G**, typing **go** on the **Command** pane, or clicking the **Run** button on the **Results** pane.

You can watch on the **Results** pane as the model follows the simulation procedure, first computing the initial conditions and then marching forward in time, from $\xi = 0$ to $\xi = 1$.

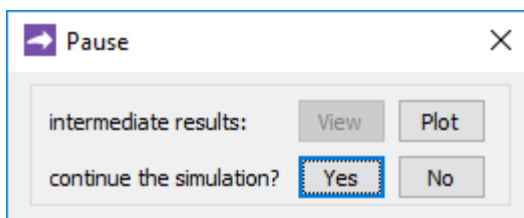
When the calculation is complete, click on the **Plot Results** button to launch **Xtplot**. This program lets you render the simulation results graphically, as xy plots or in various geochemical projections such as Piper diagrams. For two-dimensional simulations, you can render the results in map view, contouring and color mapping variables and representing groundwater flow as a vector field. The chapter [Using Xtplot](#) in this guide gives further details.

You can also have the model produce text-format or “print” results describing the chemical state at each node in the domain at certain points in time. To do so, select **Config → Output...** and then select the “print dataset” option. The program will write results at intervals in the reaction progress (ξ) set by variable “dxprint” on the **Config → Output...** dialog. When the model is complete, click on **View Results**. The print option can produce large amounts of output and, therefore, is by default turned off.

After finishing a simulation, you can revise the model configuration and run it again. When the new simulation is complete, any instances of **Xtplot** running on your computer will automatically update their plots to reflect the latest results. You can open several **Xtplot** windows on your computer at the same time, allowing you to view results plotted in different ways.

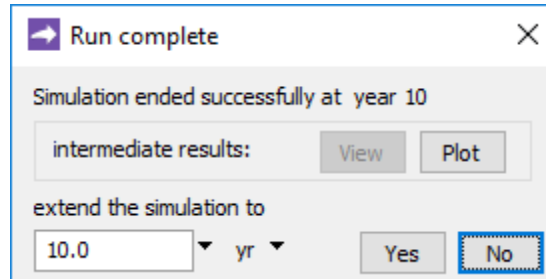
By clicking on **Run → Go Initial** or typing `go initial`, you cause the program to evaluate the initial conditions without time marching through the simulation. Once done, you can render the initial conditions with **Xtplot**, as you could the results of any simulation. This feature is handy because it lets you verify that the model has been configured to give the correct flow field, medium properties, initial fluid chemistry, and so on, without waiting for the entire simulation to complete.

Depending on the number of nodal blocks in a simulation and the number of time steps required to complete it, the calculation may take just a few seconds, or many hours. For lengthy simulations, you can interrupt a run and use **Xtplot** to examine the results of the partially completed simulation. When you press **Ctrl+Break** or click on **Run → Kill**, the simulation pauses and presents a dialog box



Clicking on **Plot** lets you plot the results to this point in the simulation. If you have selected the “print dataset” option on the **Config → Output...** dialog, clicking on **View** lets you examine the results to this point in the simulation. You can choose to continue the simulation from the current point by clicking **Yes**, or terminate it by clicking **No**.

Once the simulation is complete, the program gives you the option of extending it beyond the specified ending time. After taking the last time step, a dialog appears.



You can view the simulation results by clicking on **Plot**, or, if you have selected the “print dataset” option on the **Config → Output...** dialog, clicking on **View**. Select **No** to terminate the simulation as specified, or **Yes** to extend the simulation past the original endpoint. In the latter case, you specify a new simulation endpoint.

Using X1t

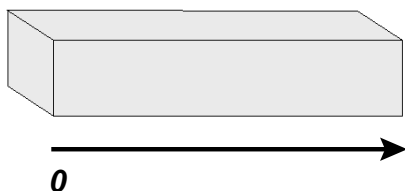
X1t is the program in the Xt package that models reactive transport in one dimension. It is in many ways similar to the **X2t** program, which models reactive transport in two dimensions. A number of concepts common to **X1t** and **X2t** are discussed in the previous chapter of this guide, [Modeling Overview](#). You should, therefore, read the previous chapter before beginning this one, and read this chapter before beginning to work with **X2t**.

The *GWB Reference Manual* contains additional details of the various commands you can use to configure **X1t**.

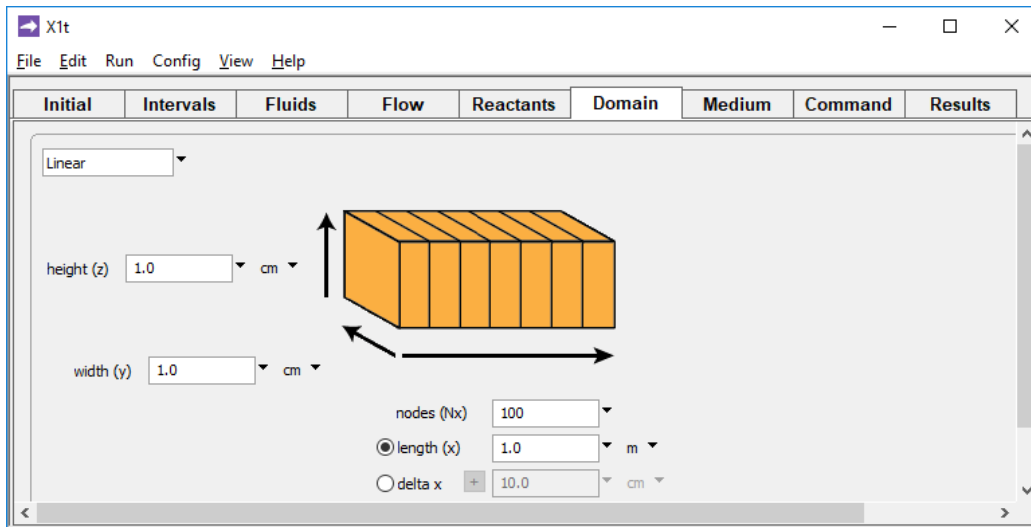
3.1 Defining the domain

You can configure **X1t** in terms of either a linear, radial, or spherical domain. The program divides the domain, regardless of type, into N_x nodal blocks, which can be evenly spaced or of varying length.

A linear domain extends from $x=0$ at its left side to $x=L$ at the right.



For a linear domain, you can set the length L and number of nodal blocks N_x on the **Domain** pane, or with the `length` and `Nx` commands. If, for example, you set a domain 1 km in length divided into 100 nodal blocks

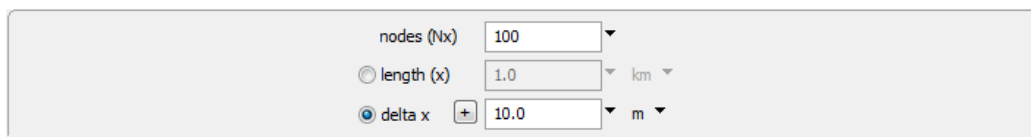


the program will set the length Δx of each block to 10 m. The commands

```
length = 1 km
Nx = 100
```

do the same thing. The program recognizes mm, cm, m, km, in, ft, and mi as units of length.


You can alternatively define Δx and let the program figure the domain length L . Setting Δx to 10 m and N_x to 100,

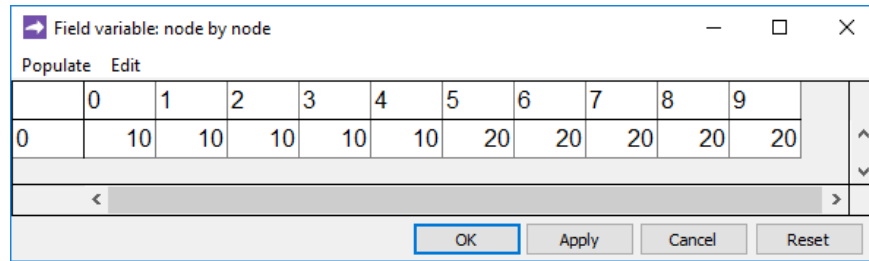


is equivalent to the previous example, in which L is 1 km. The commands

```
deltax = 10 m
Nx = 100
```

configure the same domain.

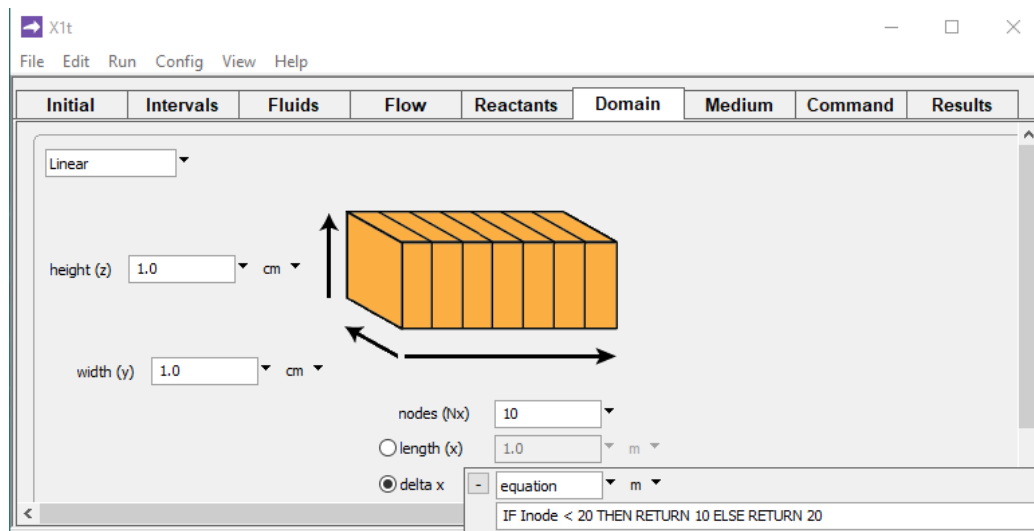
X1t carries Δx as a field variable (see the [Heterogeneity](#) appendix to this guide), so you can define a domain in which the length Δx of the nodal blocks varies with position. To do so, click on the  next to the “delta x” label on the **Domain** pane. As an example, choose “node by node” from the pulldown list and click “Edit...” to bring up an editor



or enter a command such as

```
deltax = {10 10 10 10 10 20 20 20 20 20} m
```

As a second example, choose “equation” from the list and type in a simple string for the program to interpret.



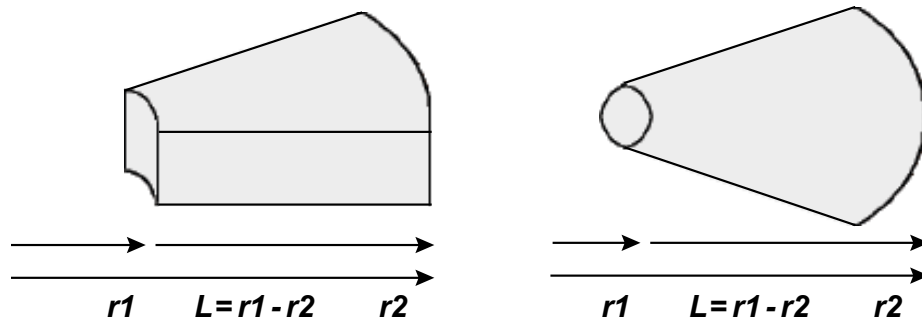
The command

```
deltax = eqn "IF Inode < 20 THEN RETURN 10 ELSE RETURN 20" m
```

is equivalent.

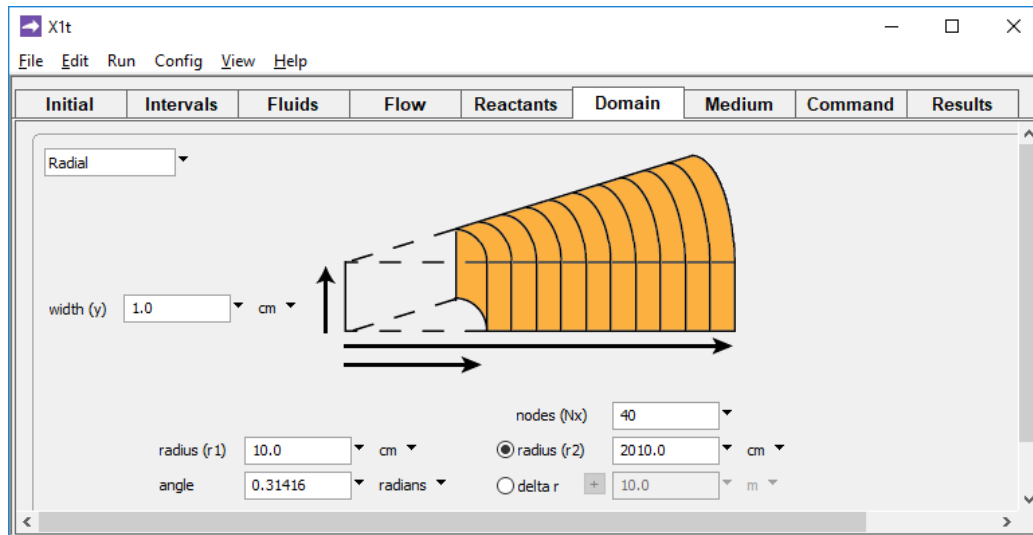
You can set the width and height of the nodal blocks on the pane, or with the `width` and `height` commands. Most commonly, however, an **X1t** simulation is configured so the settings for these values, which by default are 1 cm, do not affect the model results.

The radial and spherical domains extend from a small-radius end, at which $r = r1$, to a large-radius end, where $r = r2$.



The length L of a radial or spherical domain, then, is $r1 - r2$.

You define the coordinates $r1$ and $r2$ of a radial domain, as well as $\Delta\theta$, the angle of divergence. You set values for $r1$ and $r2$ in units of length, and $\Delta\theta$ in radians or degrees. For example, the configuration below



sets a domain composed of 40 nodes, starting 10 cm from the radial origin (e.g., 10 cm from the center of a well bore) and extending for 2000 cm; the radial angle is $\pi/10$. The default value for $\Delta\theta$ is 0.1 radians.

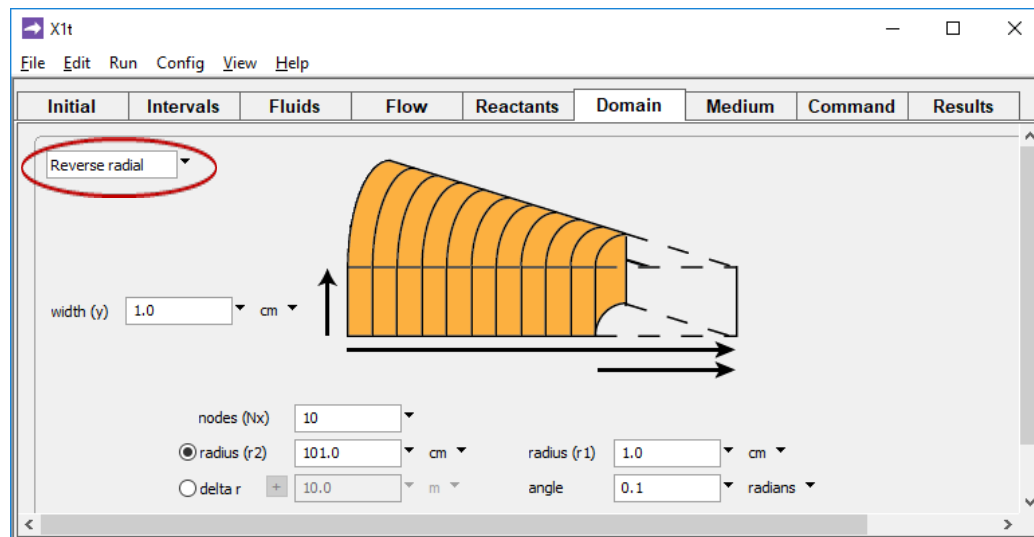
If you are working from the command line, use the `radial` (or `wedge`) command with keywords `r1`, `r2`, and `angle`. As before, the `Nx` command defines the number of nodal blocks. The commands

```
radial r1 = 10 cm, r2 = 2010 cm, angle = 0.31416
Nx = 40
```

configure the same domain as shown in the GUI above.

You can also set the node spacing Δr with the `del tar` command and let the program determine the domain length L from the values you set for N_x and $r1$, much as you would for a linear domain. As with a linear domain, you can specify Δr as a field variable to set variably spaced nodes.

Normally, the domain is oriented with the small-radius end to the left, but you can reverse its sense. To reverse the sense of a radial domain, select “Reverse radial” on the pane,



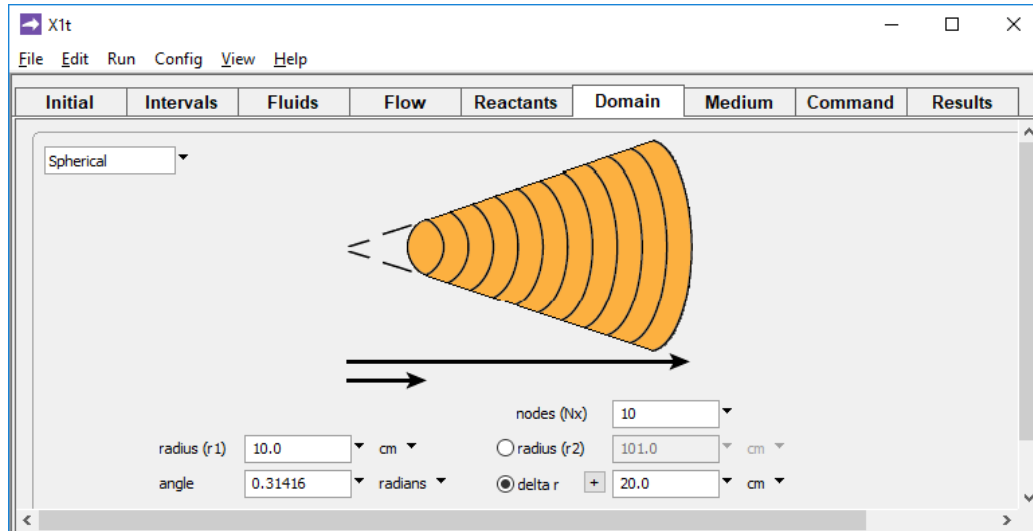
or enter the command

```
radial reverse
```

Once you have set a radial domain, you can return to linear coordinates on the pane, or with the command

```
radial off
```

You define the coordinates $r1$ and $r2$ of a spherical domain, as well as $\Delta\theta$, much the same as you do with a radial domain.



If you are working from the command line, use the `spherical` (or `spike`) command with keywords `r1`, `r2`, and `angle`. As with the radial domain, you can set values for `r1` and either `r2` or the nodal block spacing Δr . The commands

```
spherical r1 = 10 cm, angle = 0.31416
deltar = 20
```

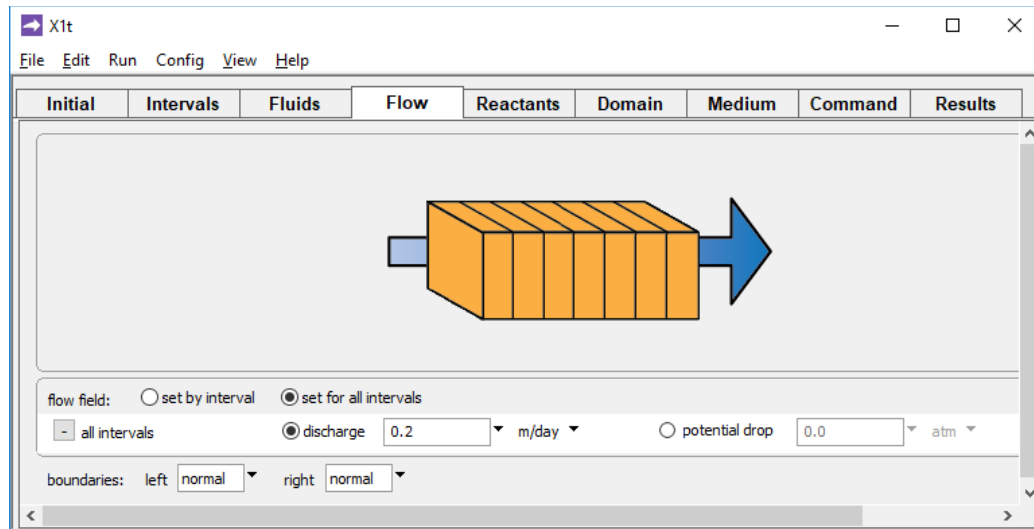
are equivalent to the **Domain** pane settings above.

Once you have set a spherical domain, you can return to linear coordinates on the **Domain** pane, or with the command `spherical off`. To reverse the sense of a spherical domain, select “Reverse spherical” on the pane, or enter the command `spherical reverse`.

3.2 Setting flow rate

You define the rate of groundwater discharge across the domain in one of two ways: by setting a value directly or by setting the driving force for flow. In the latter case, **X1t** calculates discharge from the driving force and the values calculated for permeability and fluid viscosity. **X1t** carries the flow rate in terms of specific discharge—i.e., as a volumetric flux—rather than the “true” or average linear velocity of groundwater flow, which is specific discharge divided by porosity.

You define discharge on the **Flow** pane, or from the command line using either the `discharge`, `head_drop`, or `pot_drop` command. You can set specific discharge directly in any of a variety of units. For example, the setting



sets a specific discharge of 0.2 meters per day (or, more precisely, $0.2 \text{ m}^3/\text{m}^2/\text{day}$).

The command

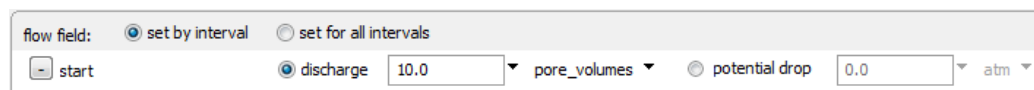
```
discharge = 0.2 m/day
```

is equivalent. The program recognizes a number of units, including mm, cm, m, km, in, ft, mi, cm^3/cm^2 , m^3/m^2 , or ft^3/ft^2 , expressed per unit time (/s, /hr, /day, /mon, /yr, or /m.y.). The default is cm/s.

Specific discharge varies slightly along the domain because of effects such as changing porosity and the thermal expansion of the fluid. In a radial or spherical domain, of course, specific discharge varies sharply along the flow path, decreasing with increasing r , because of the diverging nature of flow.

When you set discharge directly, **X1t** takes positive values to represent flow entering the left side of the domain, and negative values to represent flow on the right side.

You can also set discharge directly in terms of the number of times the medium's initial pore volume will be displaced over the course of the simulation, using units of "pore_volumes". The following setting



causes the program to flush the domain 10 times. The command

```
discharge = 10 pore volumes
```

is equivalent.

The alternative to setting discharge directly is to have **X1t** determine it over the course of the simulation from the driving force for flow you set and the values computed for permeability and viscosity. In this case, specific discharge varies over the course of the simulation in response to the evolving permeability and viscosity fields.

At any point in the simulation, specific discharge is given by Darcy's law

$$q = \frac{k}{\mu} \left(\frac{\Delta\Phi}{L} \right) \quad (3.1)$$

where q is discharge ($\text{cm}^3/\text{cm}^2/\text{s}$), k is average permeability (darcys), μ is viscosity (cp), $\Delta\Phi$ is the drop in hydraulic potential across the domain (atm), and L is the domain length (cm). Since k can change with position over the domain, **X1t** solves for q using a finite difference technique. For radial domains, the program accounts for the radial nature of flow when calculating q .

X1t computes permeability using an empirical correlation you control (see [Permeability correlation](#) in the **Modeling Overview** chapter of this guide) and estimates viscosity from the fluid's temperature and chlorinity, according to data compiled by Phillips et al. (see the appendix [Further Reading](#)).

You can specify the driving force for flow either directly, as the potential drop, in units of pressure,

The screenshot shows a software interface for setting the flow field. It has two radio buttons: 'set by interval' and 'set for all intervals', with 'set for all intervals' selected. Below this, there are two options: 'discharge' and 'potential drop'. The 'discharge' option is set to '0.0' with units 'cm/s'. The 'potential drop' option is selected and set to '0.1' with units 'atm'. There is also a checkbox for 'all intervals' which is currently unchecked.

or in terms of the drop in hydraulic head,

This screenshot is similar to the previous one, but the 'head drop' option is selected instead of 'potential drop'. It is set to '150.0' with units 'cm'. The 'discharge' option remains at '0.0' cm/s, and 'set for all intervals' is still selected.

which is expressed in units of length. In the latter case, the program computes the potential drop $\Delta\Phi$ from the head drop Δh according to

$$\Delta\Phi = \rho g \Delta h \quad (3.2)$$

assuming a fluid density ρ of 1 g/cm³ and an acceleration of gravity g of 980.1 cm/sec². If you are working from the command line, you set the driving force with the `pot_drop` or `head_drop` command.

Examples include:

```
pot_drop = 0.1 atm
head_drop = 150 cm
```

The program recognizes the units of pressure Pa, Mpa, atm, bar, and psi for potential drop, and the units of length mm, cm, m, km, in, ft, and mi for head drop; the default is atm.

If you have specified more than one reaction interval, you can set distinct values for discharge

flow field: <input checked="" type="radio"/> set by interval <input type="radio"/> set for all intervals					
<input type="checkbox"/> start	<input checked="" type="radio"/> discharge	<input type="text" value="2.0"/>	m/day	<input type="radio"/> potential drop	<input type="text" value="0.0"/> atm
<input type="checkbox"/> interval_2	<input checked="" type="radio"/> discharge	<input type="text" value="-2.0"/>	m/day	<input type="radio"/> potential drop	<input type="text" value="0.0"/> atm

or driving force for the various intervals on the **Flow** pane.

flow field: <input checked="" type="radio"/> set by interval <input type="radio"/> set for all intervals					
<input type="checkbox"/> start	<input type="radio"/> discharge	<input type="text" value="2.0"/>	m/day	<input checked="" type="radio"/> head drop	<input type="text" value="70.0"/> cm
<input type="checkbox"/> interval_2	<input type="radio"/> discharge	<input type="text" value="-2.0"/>	m/day	<input checked="" type="radio"/> head drop	<input type="text" value="-70.0"/> cm

Both options set a positive discharge for the first reaction interval and a negative discharge for the second. The commands

```
discharge start = 2 m/day, interval-2 = -2 m/day
head_drop start = 70 cm, interval-2 = -70 cm
```

do the same thing.

Finally, you can use the `discharge`, `head_drop`, and `pot_drop` commands to assign a boundary fluid, as alternatives to the `interval` command. For example, the command

```
discharge .002 m/s, fluid = my_fluid
```

sets “my_fluid” to cross into the domain along the left boundary at .002 m s⁻¹, whereas the commands

```
discharge start fluid = my_fluid1
discharge interval-2 fluid = my_fluid2
```

assign distinct fluids to simulation intervals “start” and “interval-2”.

3.3 Mass transport

X1t accounts for the transport of the chemical components in solution by molecular diffusion, hydrodynamic dispersion, and groundwater advection. The program models the combined processes of diffusion and dispersion with a “Fickian” linear law, so-called because it takes the form of Fick’s law of diffusion.

Fick’s law gives the dispersive flux q_D (mol/cm²/s) of a chemical component in solution as

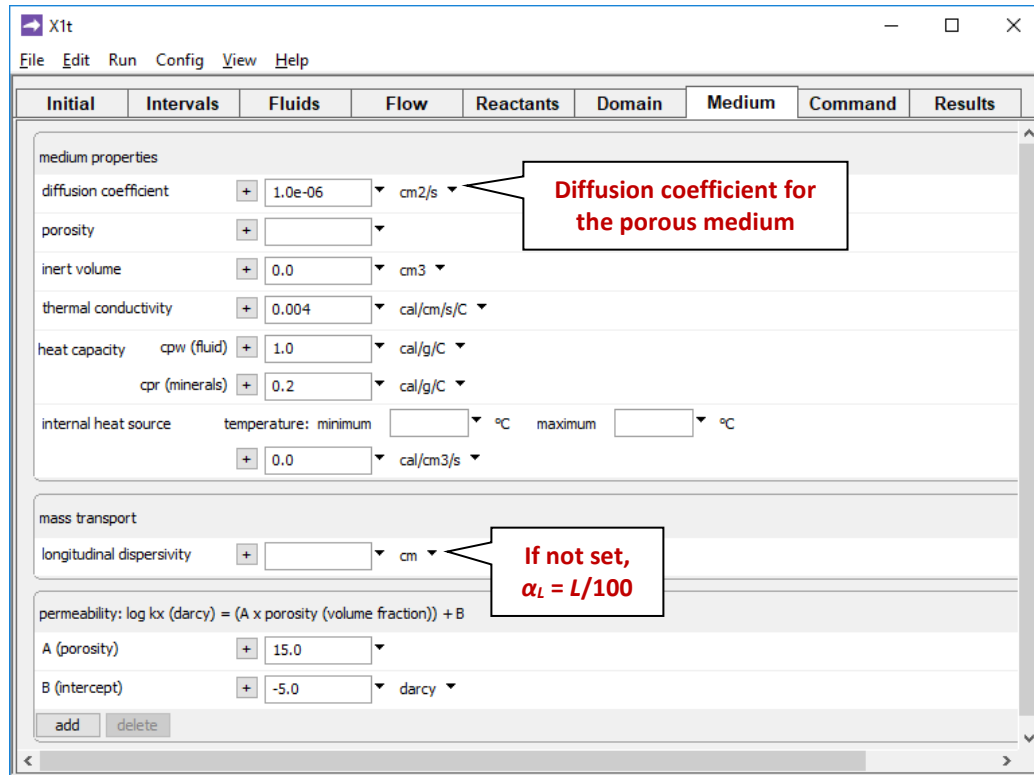
$$q_D = -\phi D \frac{\partial C}{\partial x} \quad (3.3)$$

where ϕ is sediment porosity, D is the coefficient of hydrodynamic dispersion (cm²/s), and C is the volumetric concentration (mol/cm³) of the component in question.

The coefficient of hydrodynamic dispersion is calculated as

$$D = D^* + \alpha |v| \quad (3.4)$$

where D^* is the diffusion coefficient (cm²/s), α is the dispersivity (cm), and v is the average linear groundwater velocity (cm/s). You set D^* and α on the **Medium** pane



or with the `diffusion_coef` and `dispersivity` commands. By default, the program assumes that D^* is $10^{-6} \text{ cm}^2/\text{s}$ and α is the length of the domain L divided by 100.

The advective flux q_A of a chemical component is given as

$$q_A = q C \quad (3.5)$$

where q is specific discharge and C is the component's concentration.

In solving for dispersive transport by the forward-in-time method, which **X1t** uses, the length Δt of a time step is limited by the von Neumann criterion

$$\left(\frac{2D}{\Delta x^2} \right) \Delta t \leq 1 \quad (3.6)$$

where Δx is the nodal block spacing. If the time marching procedure takes too long of a time step, the solution becomes numerically unstable. By this criterion, the maximum time step in a simulation decreases quadratically with the nodal block spacing.

Similarly, the Courant condition

$$\left(\frac{|v|}{\Delta x}\right) \Delta t \leq 1 \quad (3.7)$$

where v is fluid velocity, must be honored when solving for advective transport by the forward-in-time, upstream-weighted method. By this criterion, fluid may move no farther in the domain than the length of a nodal block.

Since **X1t** solves for dispersion and advection at the same time, it figures the limiting value for a time step according to

$$\left(\frac{|v|}{\Delta x} + \frac{2D}{\Delta x^2}\right) \Delta t \leq 1 \quad (3.8)$$

The program evaluates this equation for Δt at each node and uses the minimum of the values as the limiting time step.

3.4 Heat transfer

X1t, as discussed in the previous chapter, can trace polythermal simulations that account for heat transfer within the domain. The model accounts for the heat transfer of heat advecting groundwater, as well as heat conduction, the rate of which is given by Fourier's law

$$q_H = -K_H \frac{\partial T}{\partial x} \quad (3.9)$$

where q_H is the conductive flux (cal/cm²/s) and K_H is thermal conductivity (cal/cm/s °C). The previous chapter gives details on configuring a polythermal simulation.

Solving for heat transfer introduces an additional stability constraint on the simulation time step. The constraint is similar to that imposed when solving for mass transport, as described in the previous section. In the case of heat transfer, the program figures the limiting time step by evaluating

$$\left(\frac{|v|}{\Delta x} + \frac{2K_H}{\Delta x^2}\right) \Delta t \leq 1 \quad (3.10)$$

for Δt at each node and taking the least of the values.

3.5 Example input files

Subsequent sections of this chapter contain examples of applying **X1t** to trace various types of reactive transport simulations in one dimension. An input file corresponding to each example is available in the GWB installation directory (e.g., “\Program Files\GWB”) under the subdirectory “Script”.

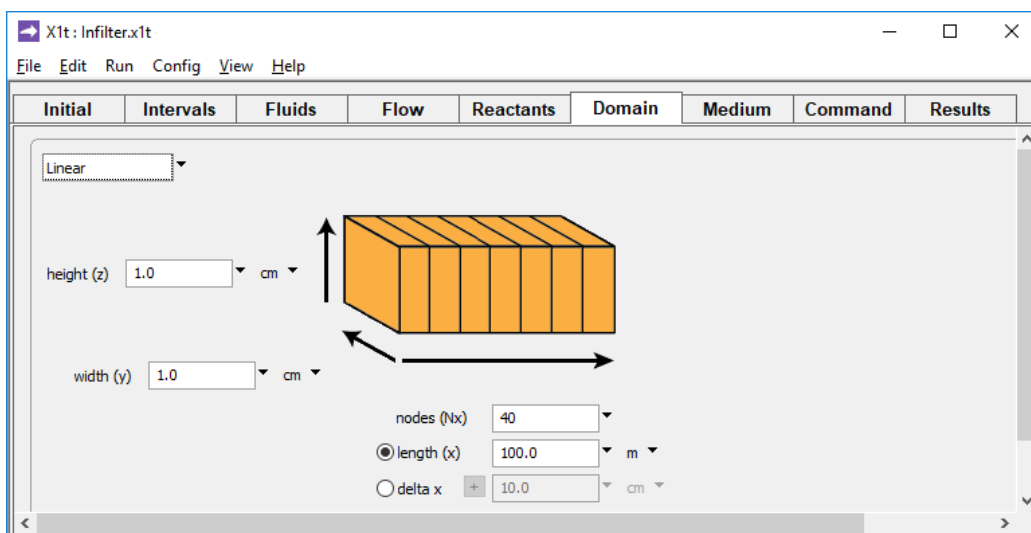
The example files are:

- | | |
|--------------------------|---|
| ▪ “Infilter.x1t” | Rainwater infiltrating a quartz aquifer |
| ▪ “Vein.x1t” | Quartz precipitation in a vein |
| ▪ “Weathering.x1t” | Weathering in a soil profile |
| ▪ “Pb_contam.x1t” | Pb contamination |
| ▪ “Chromatography.x1t” | Groundwater chromatography |
| ▪ “Steam.x1t” | Steam flood |
| ▪ “DualPorosity.x1t” | Dual porosity model |
| ▪ “Colloid.x1t” | Colloid-facilitated transport |
| ▪ “IsotopeTransport.x1t” | Stable isotope transport |

3.6 Example: rainwater infiltrating a quartz aquifer

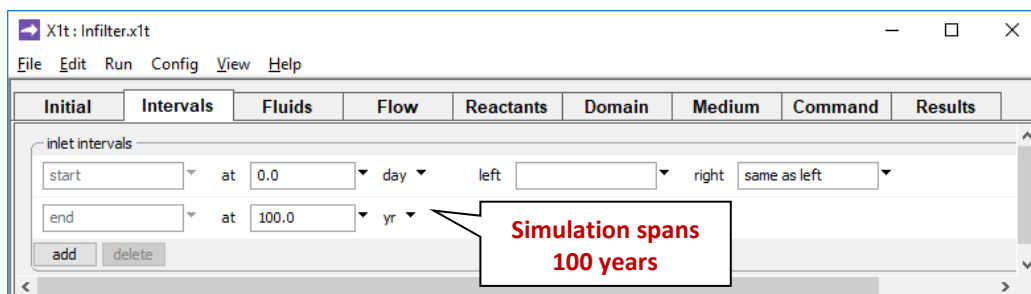
As a first example, we consider the reaction of rainwater as it infiltrates an aquifer composed of only quartz (SiO_2) grains. The water has an initial silica concentration of 1 mg/kg, somewhat less than the value of 6 mg/kg in equilibrium with quartz. As such, we expect quartz to dissolve into the water as it flows through the aquifer.

Start by double-clicking on the “Infilter.x1t” input file. When **X1t** opens, move to the **Domain** pane



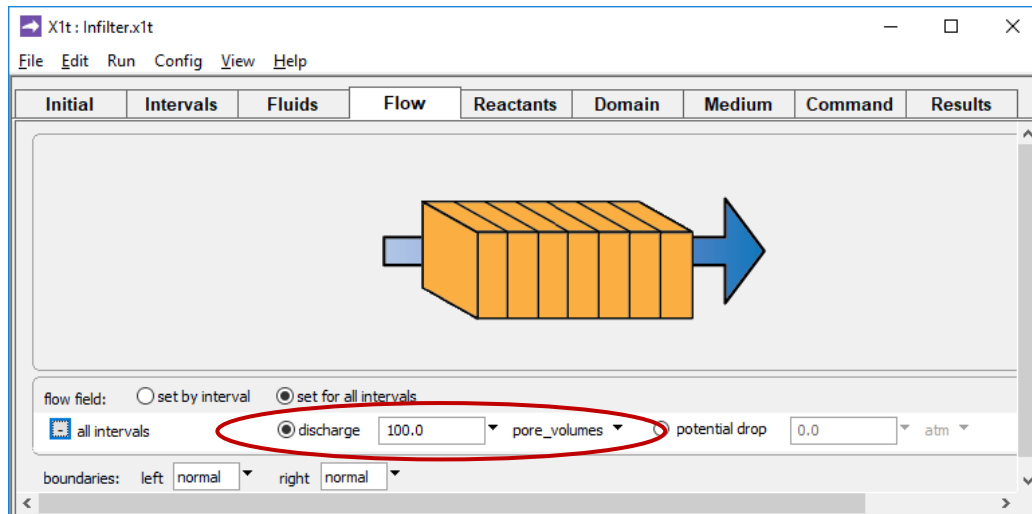
to specify a domain 100 m long, composed of 40 nodal blocks.

On the **Intervals** pane,



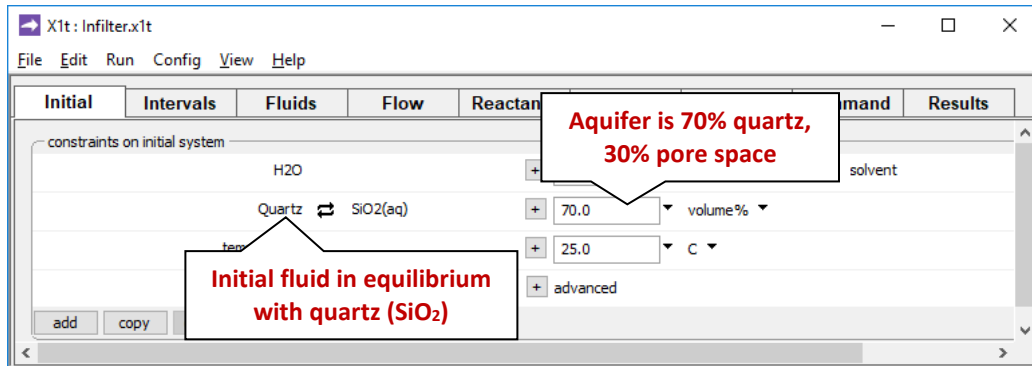
we set the model's time span to 100 years.

On the **Flow** pane



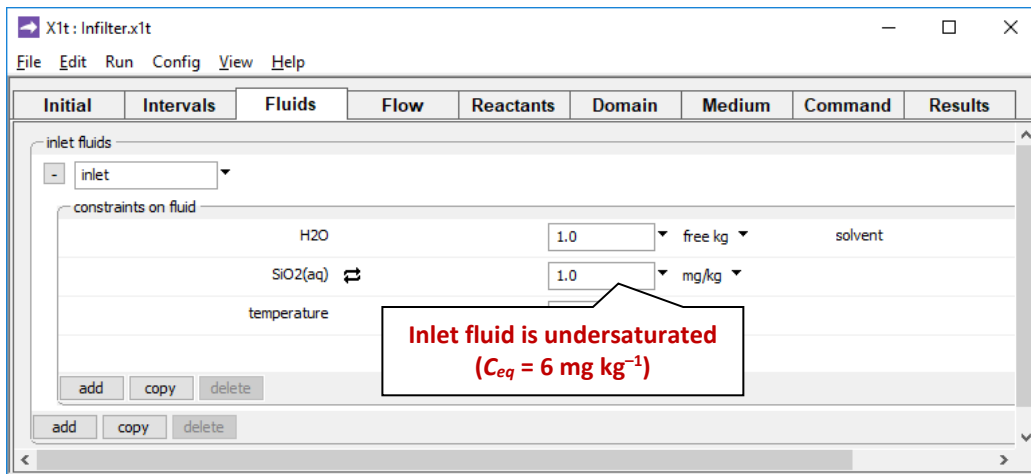
the simulation is configured to displace the pore fluid in the aquifer 100 times. Hence, the water flows at a velocity of 100 m/yr.

Continuing on to the **Initial** pane,



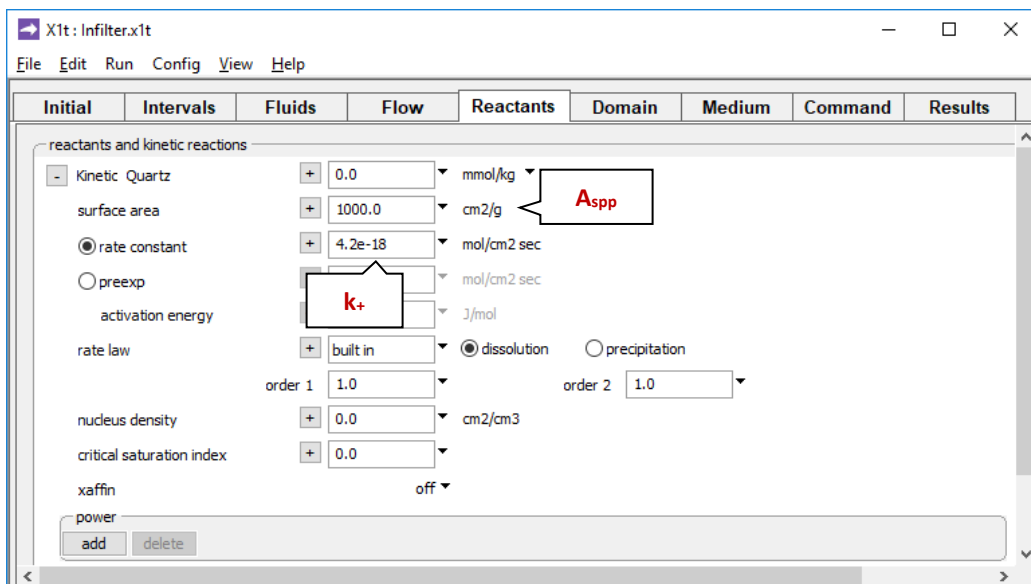
we define the initial system containing 70% quartz; the initial fluid will be in equilibrium with this mineral.

On the **Fluids** pane,



we set the inlet fluid to contain 1 mg/kg silica.

On the **Reactants** pane, we specify a kinetic rate law for quartz dissolution

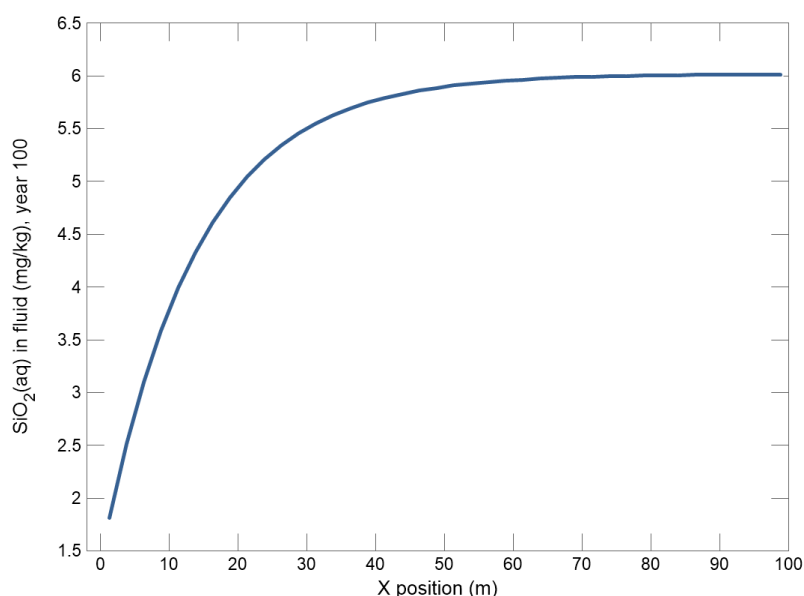


using the data of Rimstidt and Barnes (1980) and Leamson et al. (1969). The mass of quartz carried in the simulation is the amount of buffering mineral shown in the **Initial** pane, plus the mass set in the **Reactants** pane, which here is 0.

Press the **Run** button on the **Results** pane to run the model. Because of the Courant condition, the simulation will take about 4000 time steps.

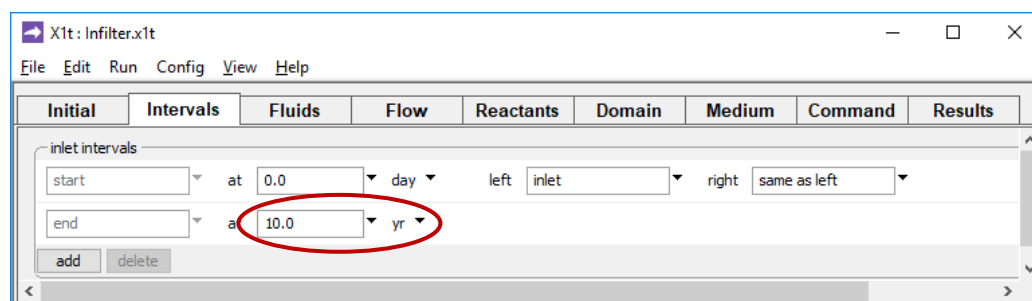
When the simulation is complete, start **Xtplot** by clicking on the **Plot Results** button. In **Xtplot**, select **Plot → XY Plot...**, then on the **X Axis** tab, next to “Display” select “X position”. Move to the **Y Axis** tab and select “Components in fluid” next to “Variable type”, then pick “SiO₂(aq)” from the list below. Under **Units**, select “mg/kg” and “linear”. Under the **Time Level** tab, select “Step 50 (100 yr)”.

Adjusting the axis ranges on the **X Axis** and **Y Axis** tabs and the labeling (**Axes and Ticks...** tab under **Format → Appearance...**), your plot should look like this:



In a similar manner, you can plot the saturation state of quartz by selecting “Mineral Saturation”, or the mineral’s reaction rate by selecting “Reactant properties” next to “Variable type” on the **Y Axis** tab.

Since we have specified that the pore fluid be displaced 100 times over the course of the simulation, we can vary the flow rate by changing the time span. For example, we can return to the **Intervals** pane to decrease the time span to 10 years.



This change has the effect of prescribing a faster groundwater flow rate of 1000 m/yr, since the pore fluid will still be displaced 100 times.

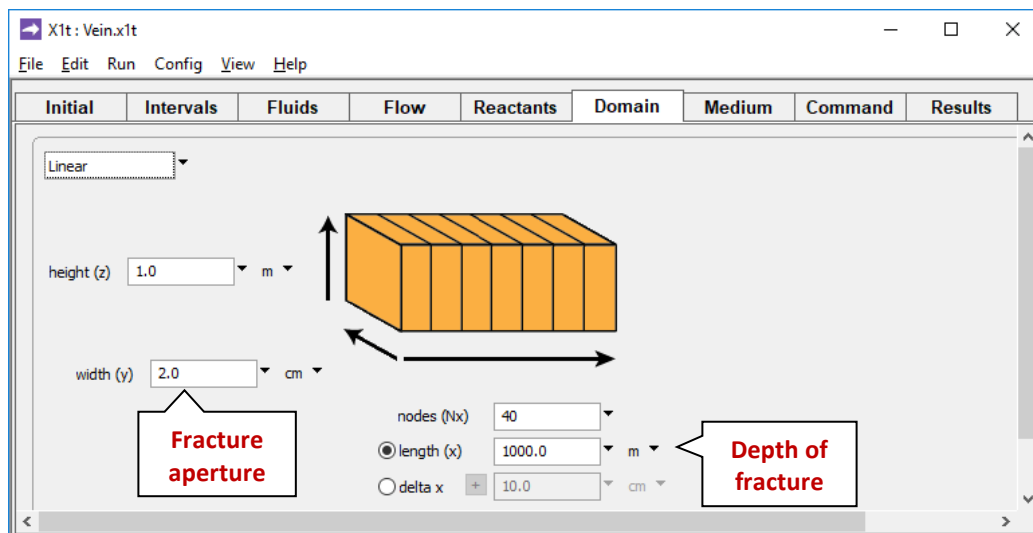
Experimenting with various velocities, you will find that when flow is slow, the fluid approaches equilibrium with quartz and reaction is slow. Under conditions of rapid flow, the fluid remains significantly undersaturated and dissolution is rapid.

It is worth noting that, unlike the simulation results, water in natural aquifers at low temperature is not commonly in equilibrium with quartz, even where quartz sand makes up a significant part of the aquifer mineralogy. This is presumably because many reactions beside quartz dissolution consume or produce aqueous silica.

3.7 Example: quartz precipitation in a vein

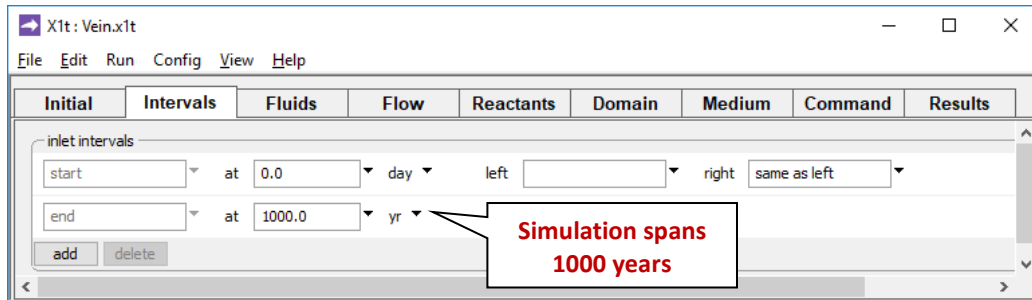
Next, we consider the problem of the kinetics of quartz precipitation in a quartz-lined vertical fracture through which hot water cools as it flows upward. The **X1t** domain in this case is the fracture itself, filled with water, hence, with an initial porosity of 100%. The x direction is vertical, along the fracture; direction y is normal to the fracture plane, which is x - z .

Start by double-clicking on the “Vein.x1t” input file. We configure the fracture’s dimensions on the **Domain** pane.



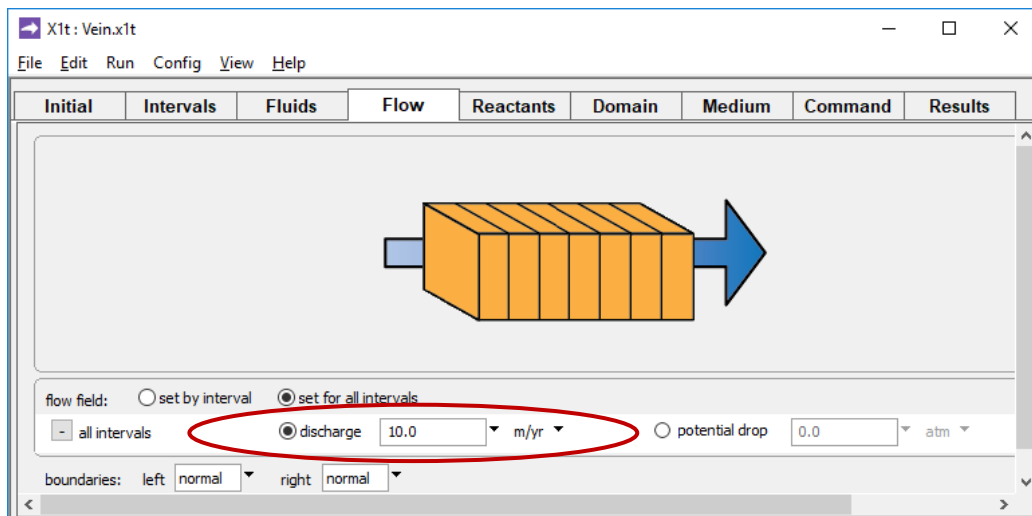
We assume the fracture extends to a depth (along x) of 1000 m and has an aperture (along y) of 2 cm. The extent of the system along z is arbitrary. We divide the medium into 40 nodal blocks.

On the **Intervals** pane,



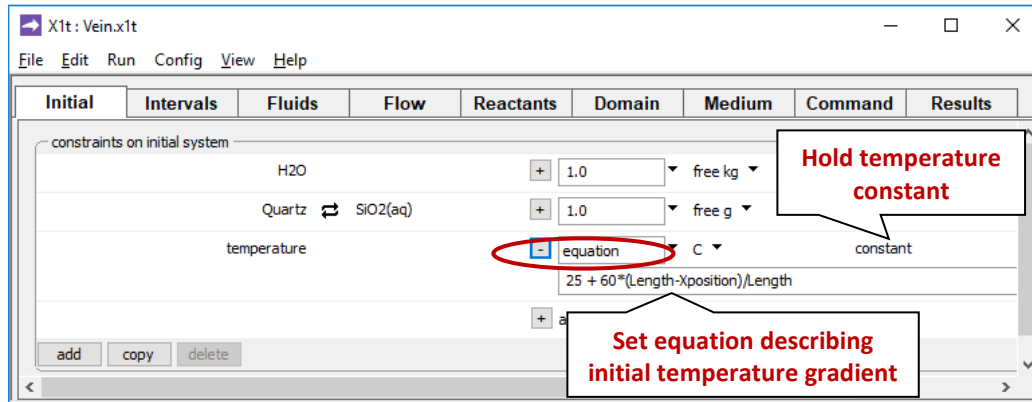
we set the model's time span to 1000 years.

On the **Flow** pane,



we specify an upward fluid discharge of 10 m/yr. Over the course of 1000 years, fluid in the fracture will be displaced 10 times.

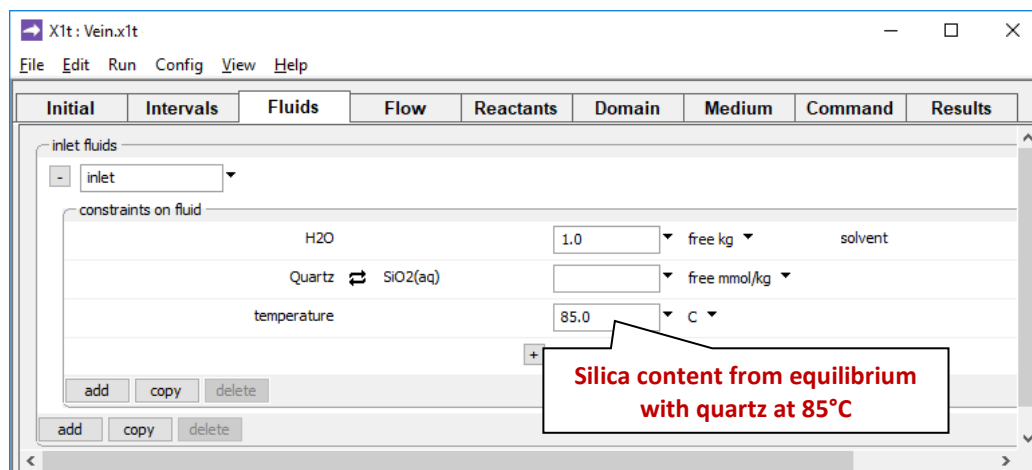
On the **Initial** pane, we prescribe a linear gradient between a surface temperature of 25°C and a basal temperature of 85°C.



Note that although we've set a heterogeneous initial temperature (as a field variable, using the equation interpreter), we specify that temperature at each point in the domain remain constant in the simulation, assuming in this simple model that conduction away from the fracture plane controls temperature within the fracture. To do so, we clicked the pulldown next to "C" and selected "constant".

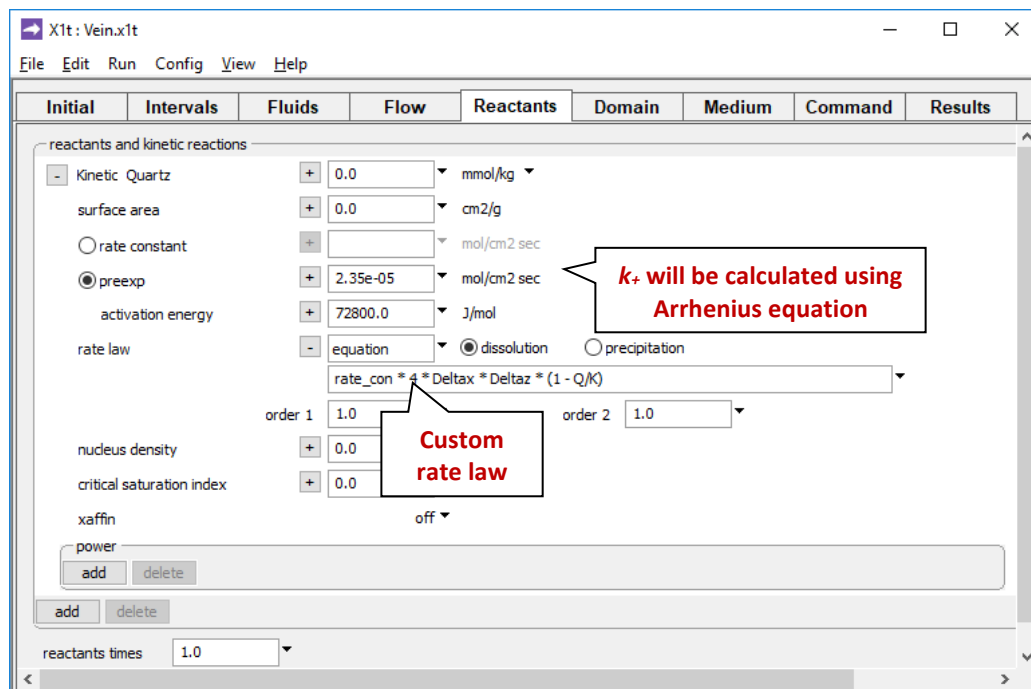
Continuing on the **Initial** pane, we specify equilibrium between quartz and the fluid's silica content at the onset of the simulation, before flow begins. To do so, we need to set only an arbitrarily small amount of quartz within each nodal block. The initial silica concentration in the model decreases along x , reflecting the decline in temperature approaching the surface.

On the **Fluids** pane,



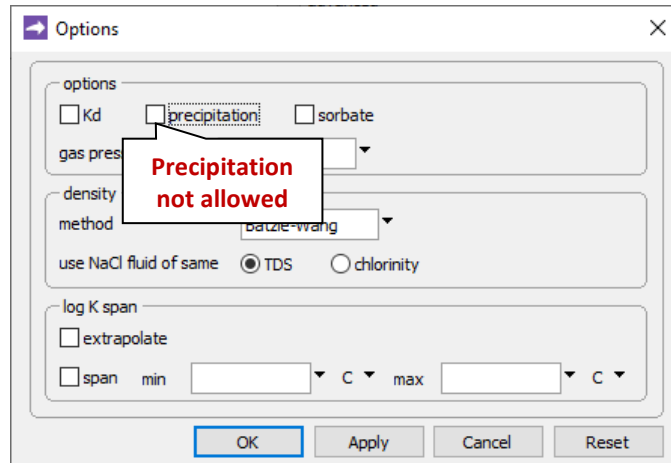
we set the silica content of the inlet fluid to equilibrium with quartz at the inlet temperature, 85°C.

We set a kinetic rate law on the **Reactants** pane. We note that the surface area within a nodal block is the area of the two fracture faces $2\Delta x\Delta y$ multiplied by the roughness of each face, which we take as 2. We define the rate constant in terms of an activation energy and pre-exponential factor, according to the Arrhenius equation, and set a custom rate law for quartz precipitation:



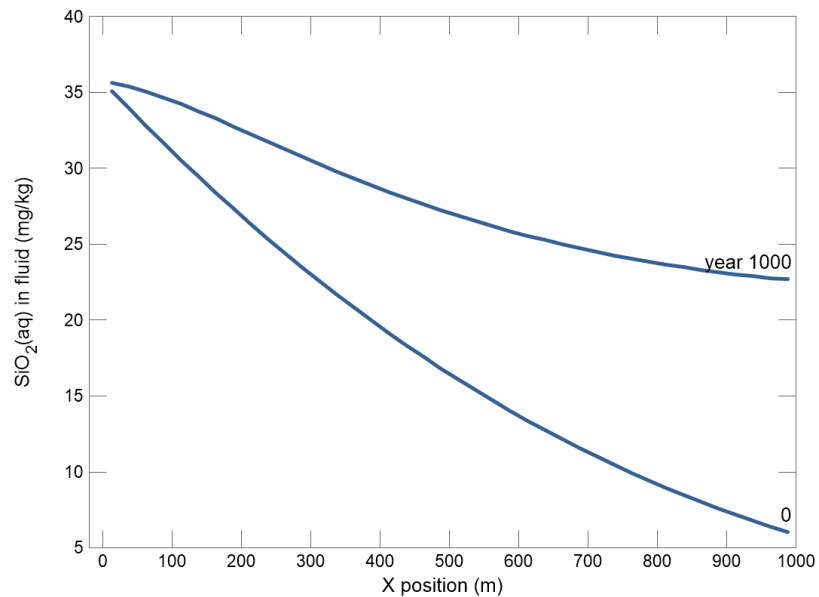
Here, we set a nominal value for specific surface area, as required by the program, although we do not use this variable.

Going to **Config** → **Options...**

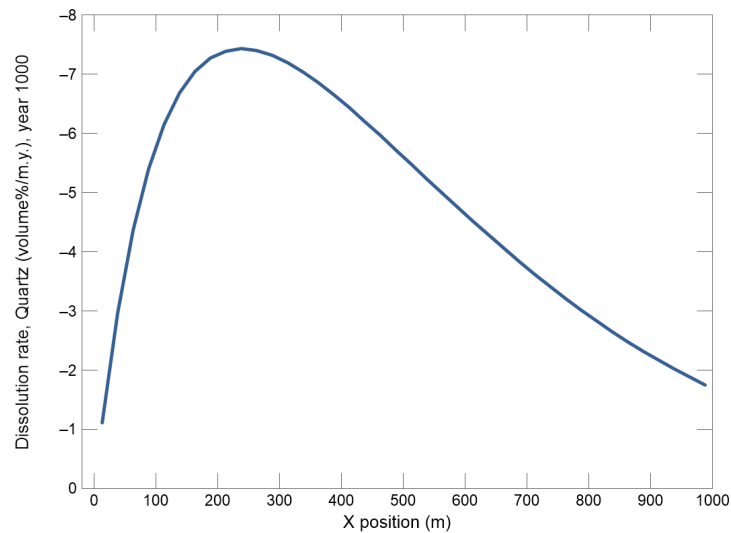


we unset the precipitation option. By doing so, we prevent other minerals, such as cristobalite and amorphous silica, from forming, although, of course, we could generalize the simulation to account for the various silica polymorphs.

Trigger the calculation by selecting **Run** → **Go**. In the modeling results, we see that, with the onset of vertical flow, silica concentration in the fracture increases from the initial equilibrium condition, becoming supersaturated. With time, the silica distribution reaches a steady state.



Quartz precipitates along the fracture, forming most quickly not near the surface, where it is most supersaturated, but at depth, where by the Arrhenius equation, the rate constant is largest.

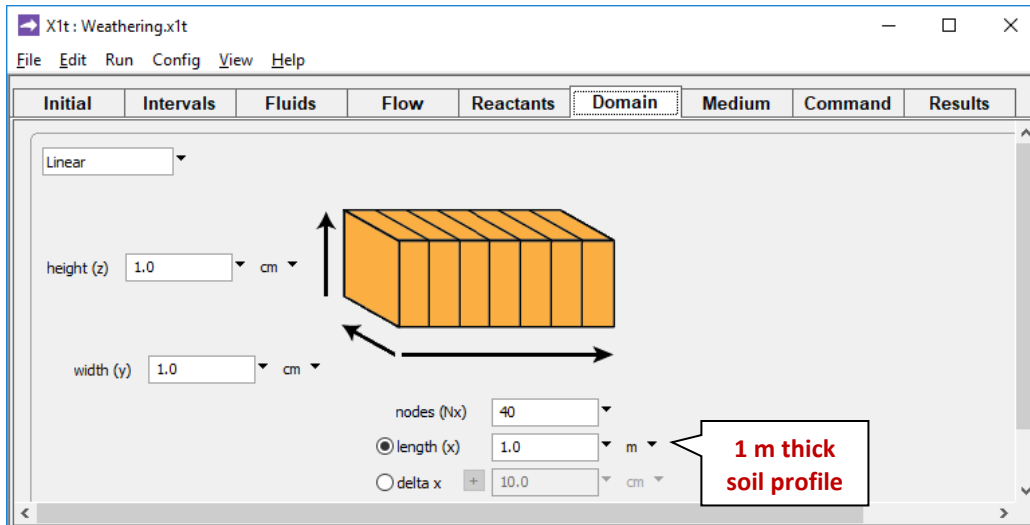


In making this plot in **Xtplot**, we select **Plot → XY Plot...**, then on the **Y Axis** tab, next to “Variable type”, select “Reactant properties”, then “Dissolution rate, Quartz”.

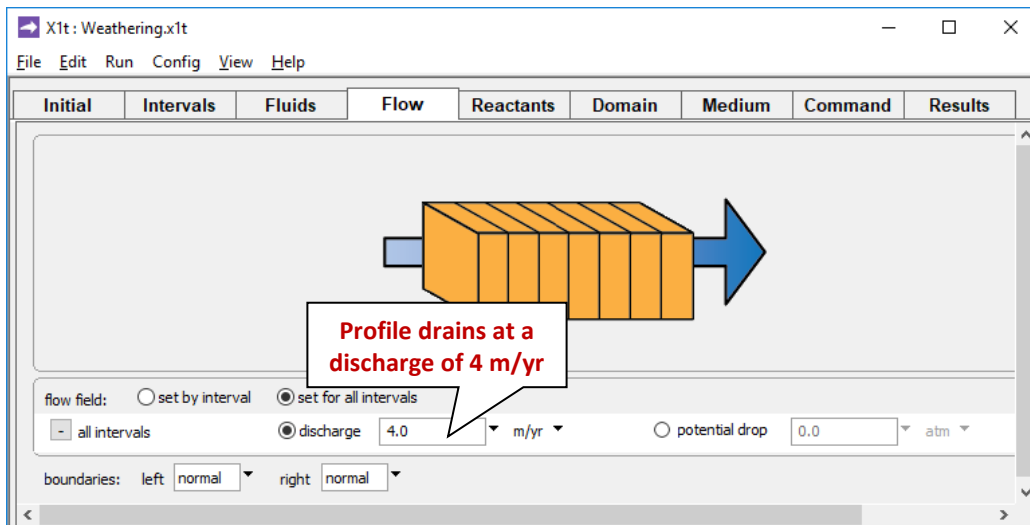
3.8 Example: weathering in a soil profile

As a further example, we construct a simple model of weathering within a soil profile. We add two new aspects to our modeling: the presence of a gas buffer within the medium and the specification of nucleation surfaces for mineral precipitation.

Start by double-clicking on the “Weathering.x1t” input file. On the **Domain** pane,

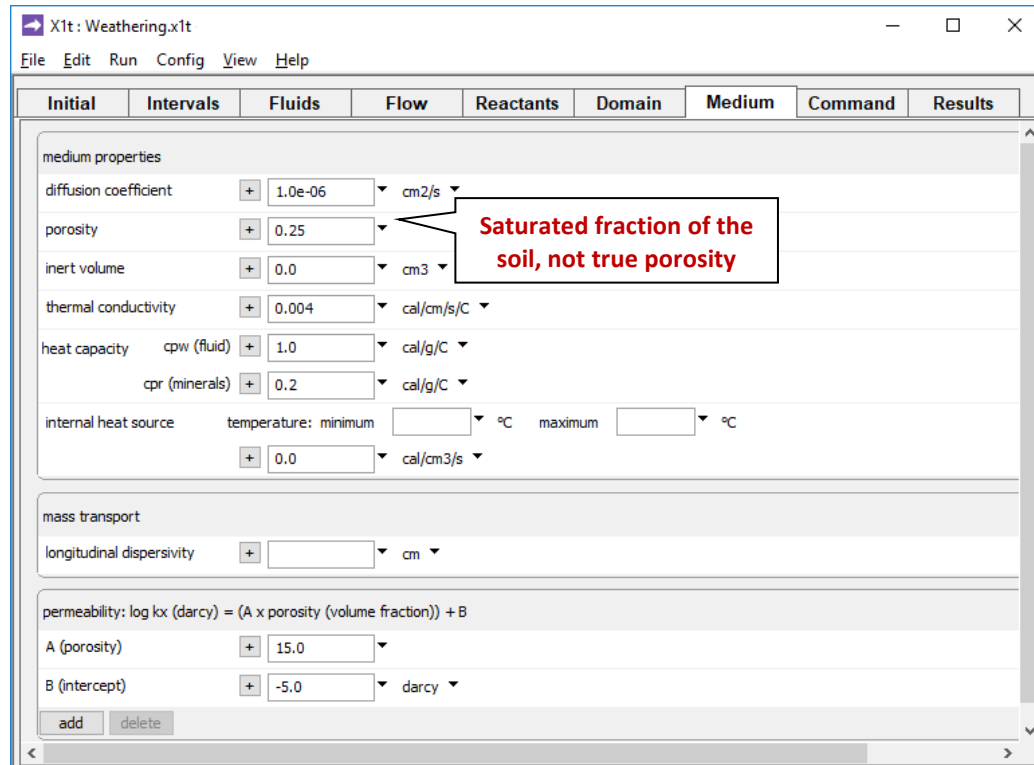


we set the profile to be 1 m thick.
On the **Flow** pane,



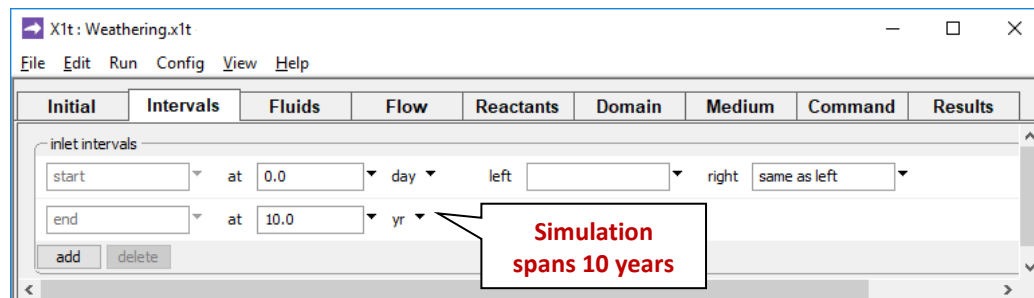
we set up the model to drain at a discharge of 4 m/yr.

On the **Medium** pane,



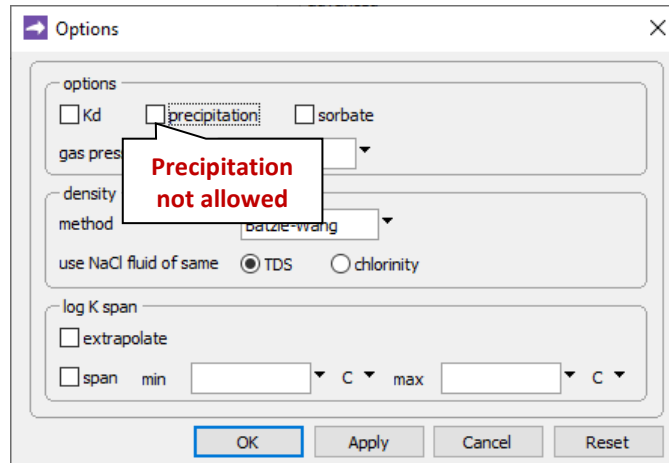
we set up our model to be 25% saturated. “Porosity” in the model represents the saturated fraction of the soil, rather than the true porosity, which includes the unsaturated fraction. The flow velocity, as we calculate from the discharge and saturated soil fraction (i.e. $v = q/\phi$), is 16 m/yr.

We specify that the simulation span 10 years, more than long enough for the calculations to assume a stationary state.



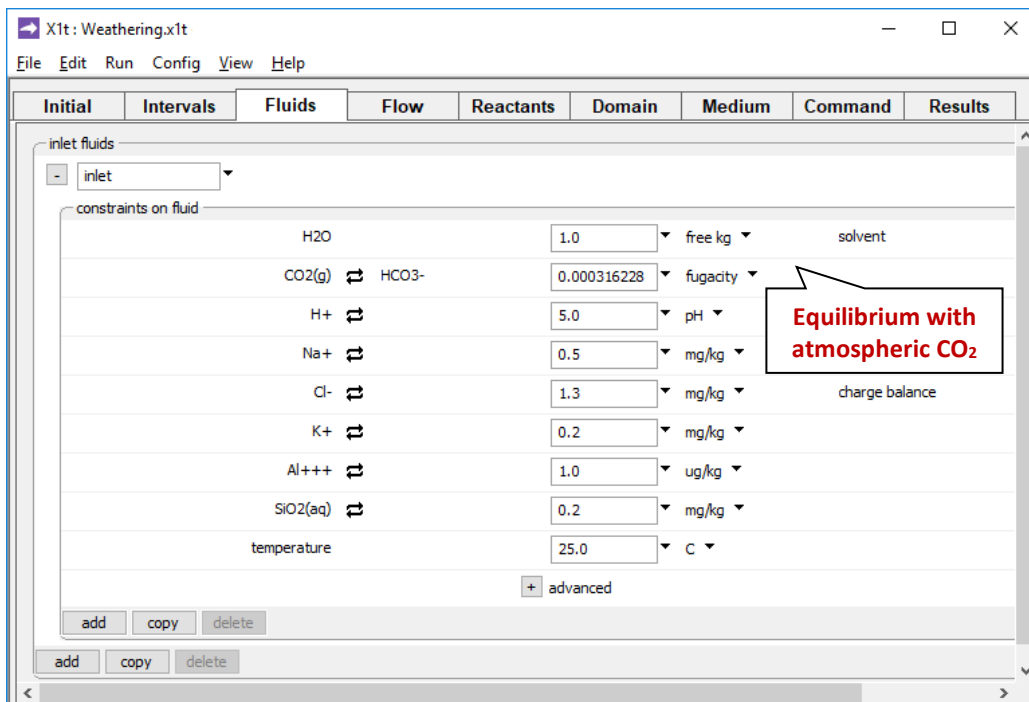
70 GWB Reactive Transport Modeling

Going to **Config** → **Options...**,



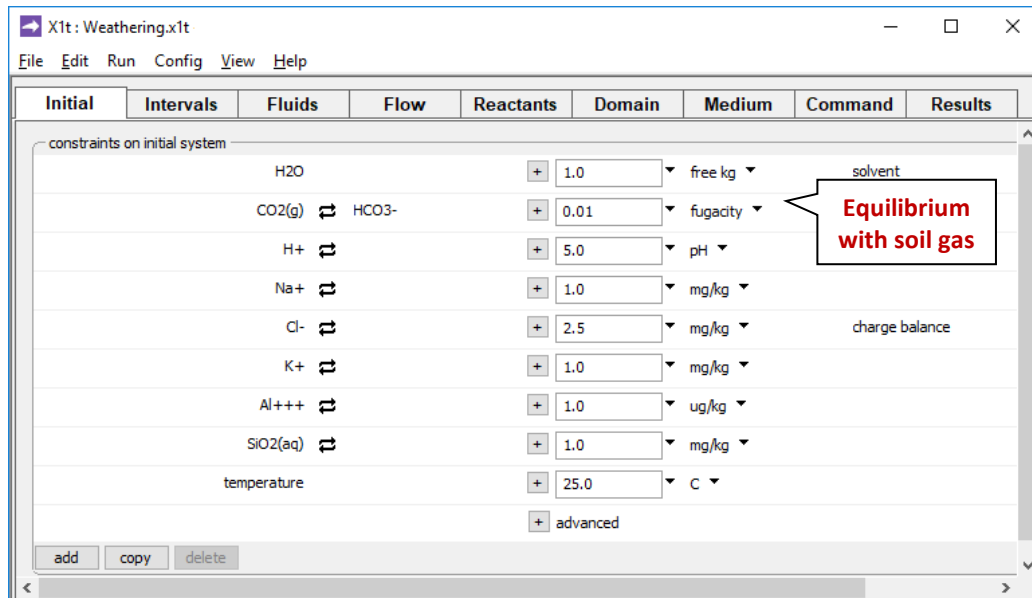
the precipitation option has been deselected. As a result, the only minerals that can precipitate are those already in the system and those for which a kinetic rate law has been set.

On the **Fluids** pane,



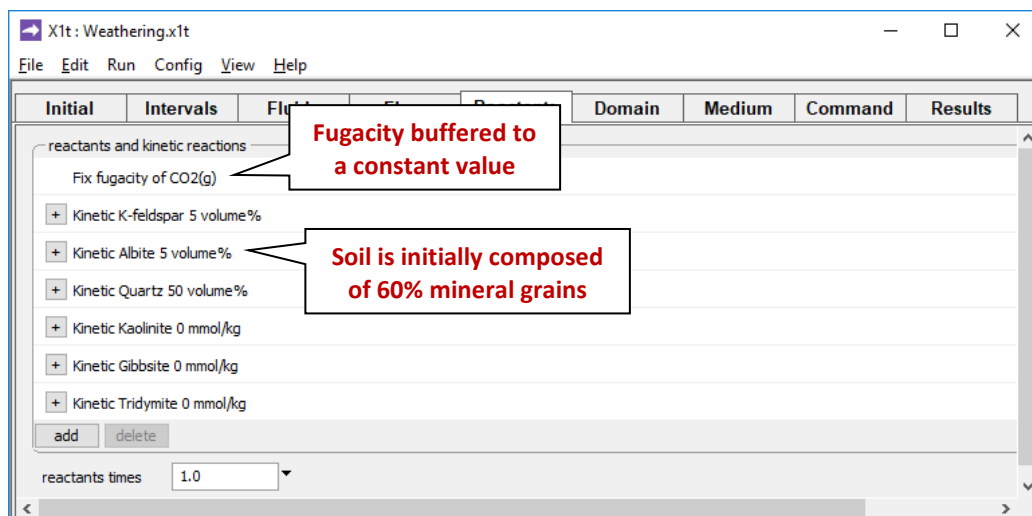
we set the inlet fluid in the simulation to be dilute water in equilibrium with atmospheric CO_2 .

On the **Initial** pane,



the fluid initially in the medium is also dilute, but is in equilibrium with the soil gas, which we take to have a CO_2 fugacity of 10^{-2} .

Setting CO_2 as a fixed fugacity reactant on the **Reactants** pane,



we cause its fugacity over the simulation to be buffered to a constant value.

72 *GWB Reactive Transport Modeling*

We set the initial soil mineralogy to a mixture of 5% potassium feldspar (KAlSi_3O_8), 5% albite ($\text{NaAlSi}_3\text{O}_8$), and 50% quartz (SiO_2), by volume. The soil, then, is composed of 25% fluid, 60% mineral grains, and 15% gas.

Next, we set kinetic rate laws describing the dissolution and precipitation of the minerals we wish to consider in the calculation. For the feldspar minerals and quartz, we take specific surface areas typical of sand grains (Leamnsen et al., 1969) and rate constants from Blum and Stillings (1995) and Rimstidt and Barnes (1980).

X1t: Weathering.x1t

File Edit Run Config View Help

Initial Intervals Fluids Flow **Reactants** Domain Medium Command Results

reactants and kinetic reactions

Fix fugacity of CO2(g)

- Kinetic K-feldspar + 5.0 volume%

surface area + 1000.0 cm2/g

● rate constant + 3.0e-17 mol/cm2 sec

○ preexp + mol/cm2 sec

activation energy + J/mol

rate law + built in ● dissolution ○ precipitation

order 1 1.0 order 2 1.0

nucleus density + 0.0 cm2/cm3

critical saturation index + 0.0

xaffin off

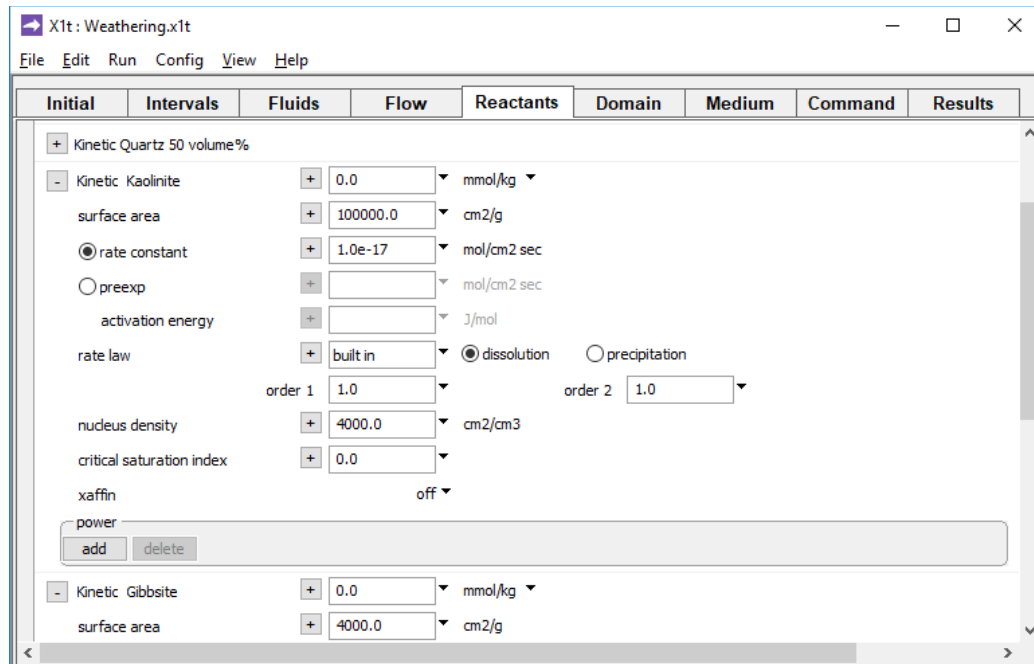
power

add delete

- Kinetic Albite + 5.0 volume%

surface area + 1000.0 cm2/g

As a final step, we set rate laws for kaolinite [$\text{Al}_2\text{Si}_2\text{O}_5(\text{OH})_4$], gibbsite [$\text{Al}(\text{OH})_3$], and tridymite (SiO_2). We take the rate constants for kaolinite and gibbsite from Nagy (1995) and estimate for tridymite from the data of Rimstidt and Barnes (1980). We further assume specific surface areas of $10^5 \text{ cm}^2/\text{g}$ for the kaolinite (Carrol and Walther, 1990), $4000 \text{ cm}^2/\text{g}$ for the gibbsite (Nagy and Lasaga, 1992), and $1000 \text{ cm}^2/\text{g}$ for tridymite.

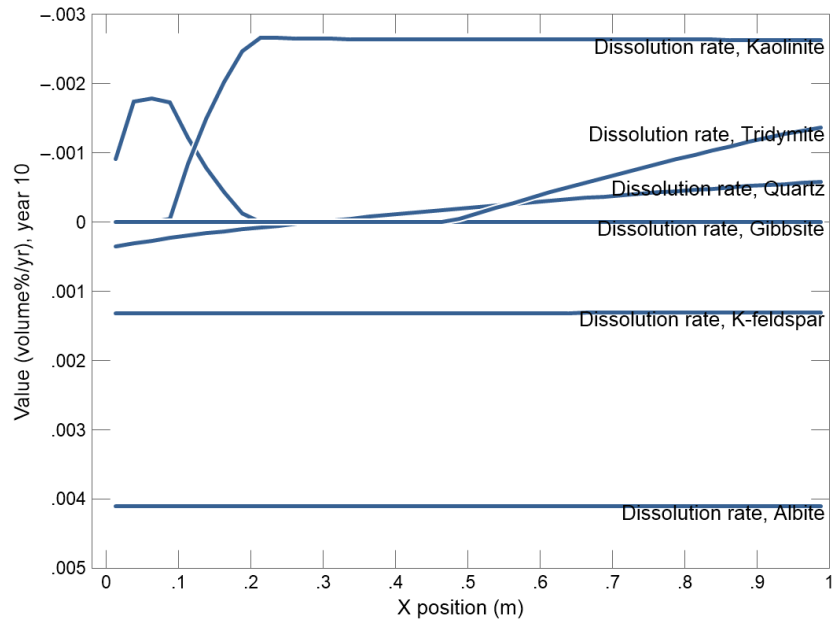


Since the latter three minerals don't exist initially in our simulation, we are faced with the added complication of describing their nucleation. Unfortunately, we have little guidance in this matter. Since sheet silicates grow readily on other mineral surfaces (by heterogeneous nucleation; Nagy, 1995), we set rather arbitrary nucleation areas of 4000 cm^2/cm^3 fluid. For comparison, the total internal surface areas of typical soils commonly fall in the range 20,000–30,000 cm^2/cm^3 bulk volume (White, 1995).

Trigger the calculation by selecting **Run → Go**. Move to the **Results** pane and click on **Plot Results** and graph the reaction rates of the various minerals as a function of position in the soil profile. Select **Plot → XY Plot...**, then on the **X Axis** tab, next to "Variable type", select "X position". Move to the **Y Axis** tab and select "Reactant properties" next to "Variable type", then choose the variables. **X1t** reports mineral dissolution rates, but by selecting "Reverse axis" on the **Y Axis** tab, you can generate a plot in which mineral precipitation increases upward.

74 *GWB Reactive Transport Modeling*

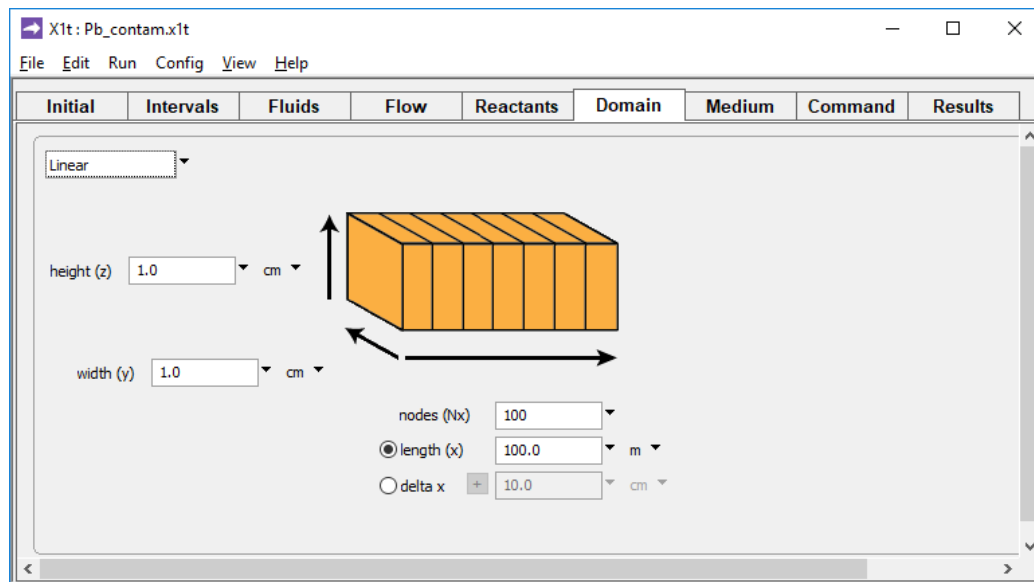
You should be able to produce a plot that looks like:



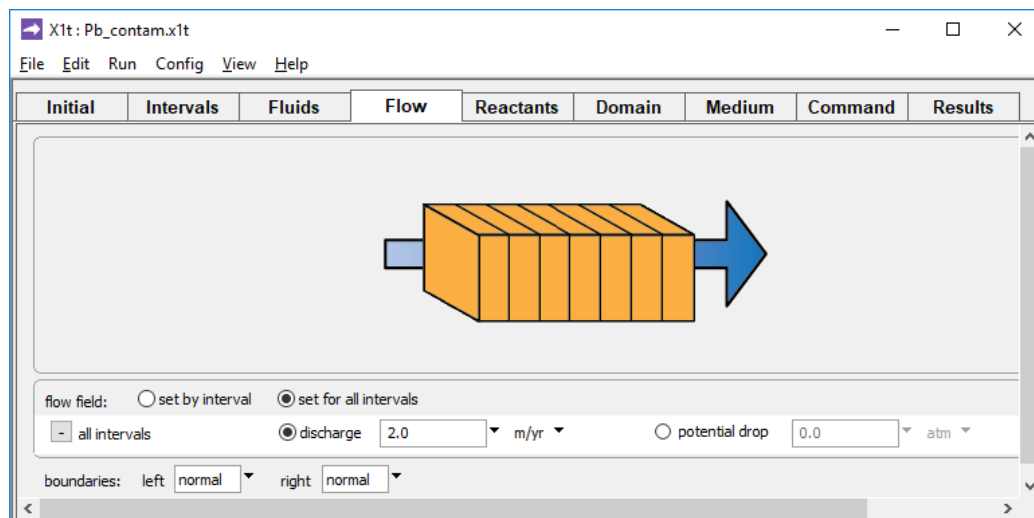
According to this diagram, K-feldspar and albite dissolve throughout the profile, in response in part to CO_2 supplied by the soil gas. Near the top of the profile, quartz dissolves as well, and gibbsite and kaolinite precipitate. Deep in the profile, kaolinite, tridymite, and quartz precipitate.

3.9 Example: Pb contamination

In this example, we model the contamination by dissolved Pb of an aquifer containing a sorbing mineral, and the effects of attempting to remediate the aquifer. Double-click on the “Pb_contam.x1t” input file to get started. On the **Domain** pane,



we set the length of the aquifer to 100 m and divide the domain into 100 nodal blocks. On the **Flow** pane,



we set a discharge of 2 m/yr.

76 *GWB Reactive Transport Modeling*

On the **Medium** pane,

The screenshot shows the 'Medium' pane of the X1t software. The 'medium properties' section includes fields for diffusion coefficient (1.0e-06 cm²/s), porosity (0.3), inert volume (0.0 cm³), thermal conductivity (0.004 cal/cm/s/C), heat capacity (cpw (fluid) 1.0 cal/g/C, cpr (minerals) 0.2 cal/g/C), and internal heat source (0.0 cal/cm³/s). The 'mass transport' section includes longitudinal dispersivity (10.0 cm) and permeability parameters (A (porosity) 15.0, B (intercept) -5.0 darcy). Callouts indicate $\phi = 30\%$ porosity and $\alpha_L = 10$ cm.

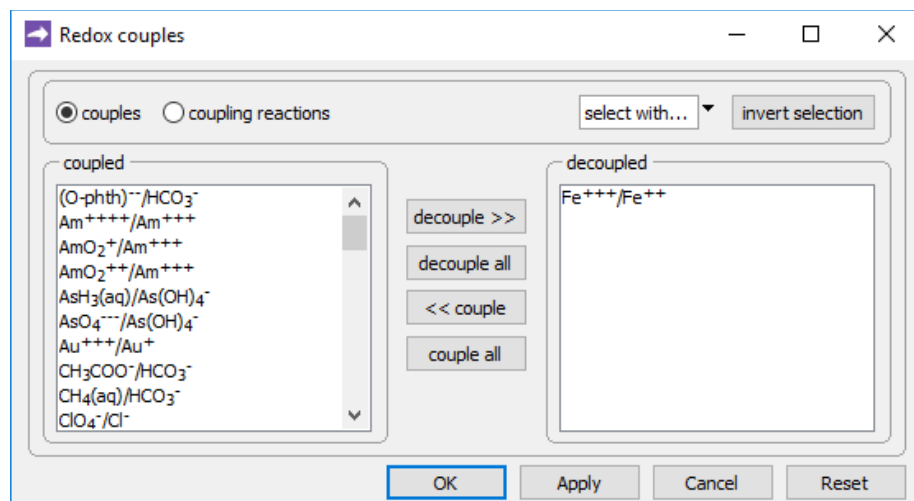
we set values for the aquifer's porosity and dispersivity.

On the **Intervals** pane,

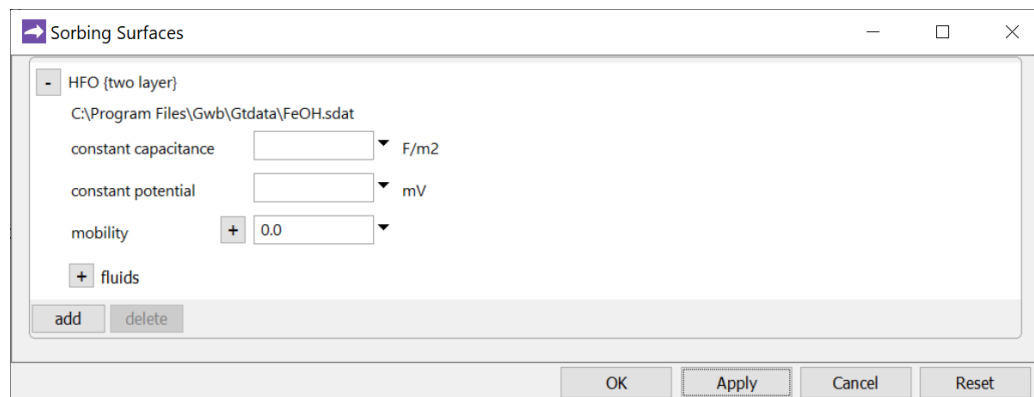
The screenshot shows the 'Intervals' pane of the X1t software. The 'inlet intervals' section includes fields for start (0.0 yr), elution (10.0 yr), and end (50.0 yr). Callouts indicate 'Metal-contaminated water enters for 10 years' and 'Clean water flushes for next 40 years'.

we let the simulation run over two reaction intervals for a total of 50 years. During the first 10 years, metal-bearing water contaminates the aquifer. Then, over a period of 40 years, it is flushed with clean water.

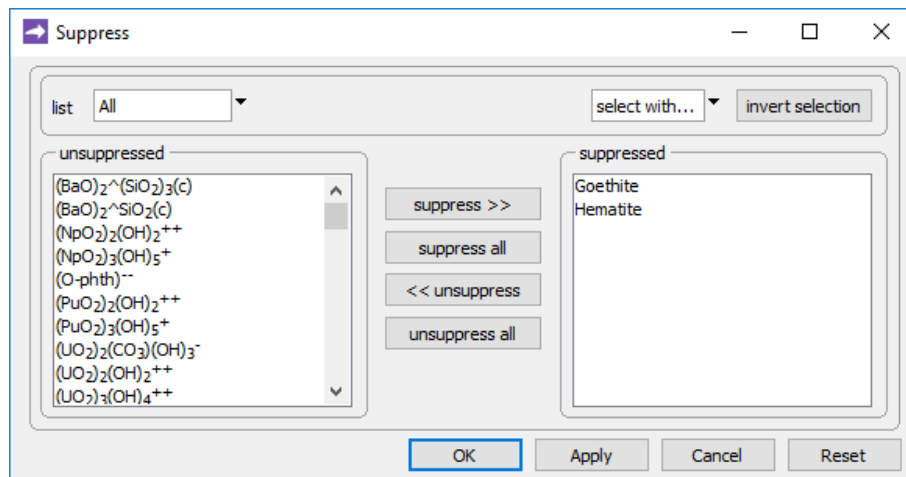
We use the two-layer model of surface complexation, as presented by Dzombak and Morel (1990), to model metal sorption and desorption. To invoke the model, we start by disabling the ferric-ferrous redox couple from the **Config → Redox Couples...** dialog.



Next, we go to **File → Open → Sorbing Surfaces...** to read in the Dzombak and Morel reaction dataset.

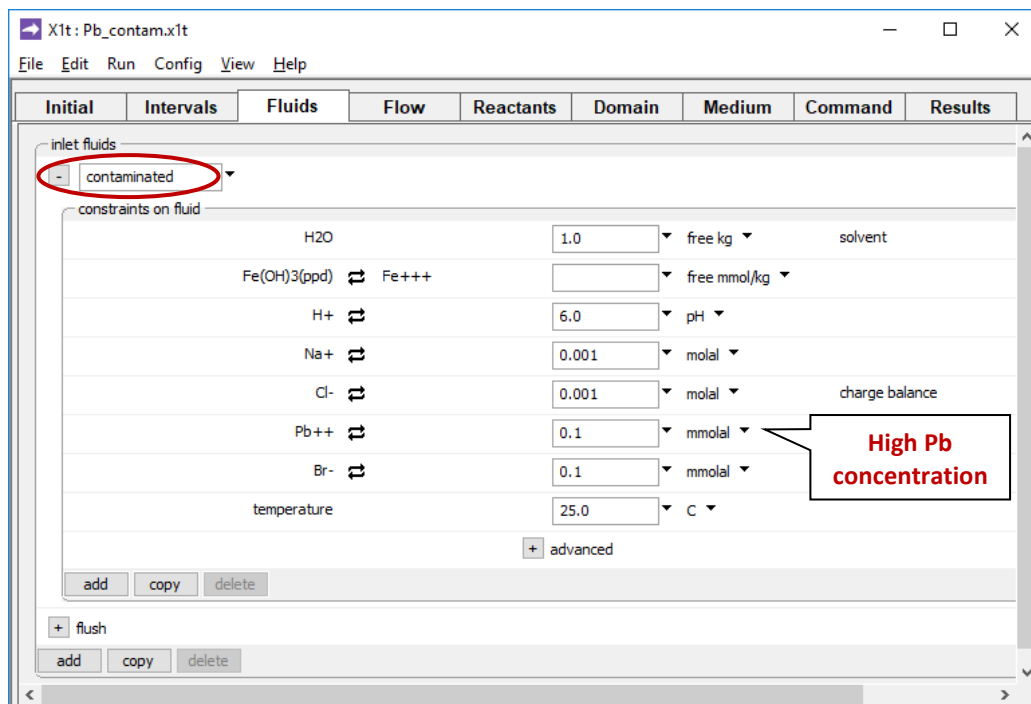


From the **Config** → **Suppress...** dialog,



we suppress the most stable ferric minerals, to allow the iron to remain in ferric hydroxide form.

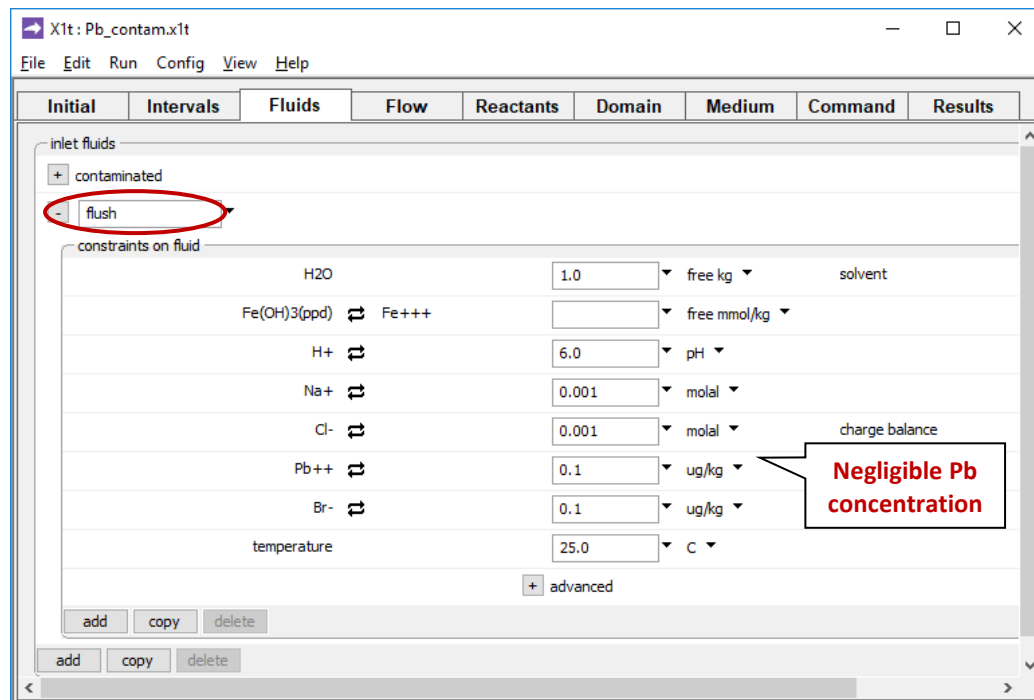
On the **Fluids** pane,



we set the contaminated fluid to a dilute water in equilibrium with ferric hydroxide [Fe(OH)₃, a proxy for hydrous ferric oxide], containing 0.1 mmolal inorganic Pb (about

21 mg/kg). For comparison, the drinking water standard (the mcl, or “maximum contaminant level”) for Pb is 15 $\mu\text{g/kg}$. The water also contains Br in equal molal concentration with the Pb. Since, according to the surface complexation model, Br does not sorb, it serves in our calculation as a conservative tracer.

Next, we set the flushing fluid,



an uncontaminated water containing almost no lead or bromide.

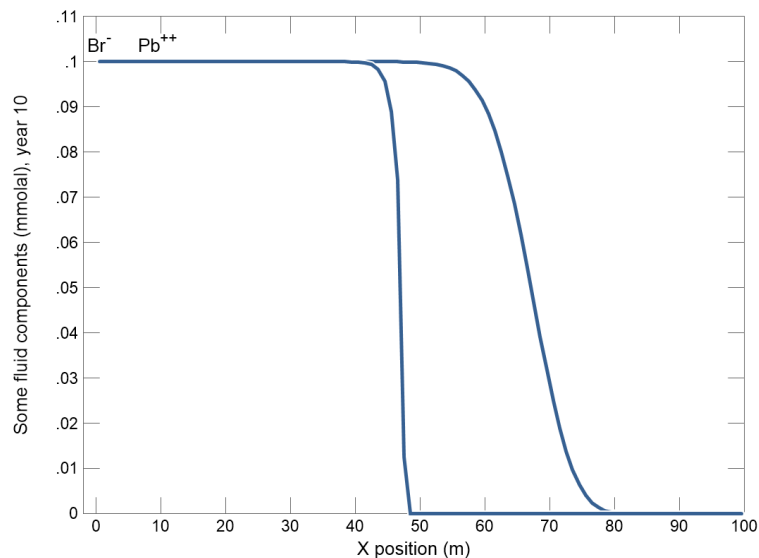
On the **Initial** pane,

Species	Initial Amount	Unit	Role
H ₂ O	1.0	free kg	solvent
Fe(OH) ₃ (ppd) ↔ Fe ⁺⁺⁺	0.01	volume%	
H ⁺ ↔	6.0	pH	
Na ⁺ ↔	0.001	molal	
Cl ⁻ ↔	0.001	molal	charge balance
Pb ⁺⁺ ↔	0.1	ug/kg	
Br ⁻ ↔	0.1	ug/kg	
temperature	25.0	C	

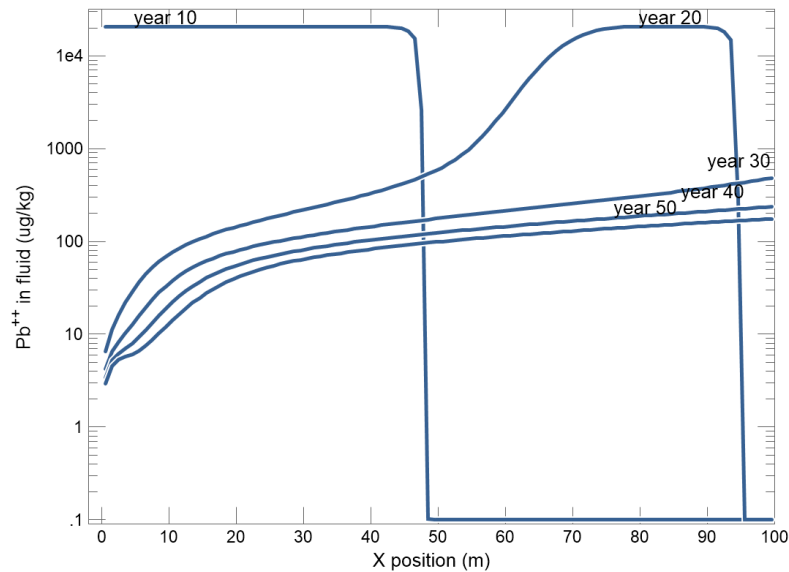
Buttons: add, copy, delete

the aquifer, which includes a small amount of ferric hydroxide, contains an initially uncontaminated groundwater of the same composition as the flushing fluid.

Once you have run the model (click **Run** on the **Results** pane), you can examine the results with **Xtplot** (click on **Plot Results**). Plotting the Pb and Br content of the fluid in the domain versus X position for the first 10 years of the model, we see that Br passes along the aquifer more quickly than Pb, since the Pb is retarded as it sorbs to the ferric hydroxide.



On the **Y Axis** pane, change the “Display” pulldown to “One value at several times” and select Pb⁺⁺, then change the scale to “log”. Select multiple steps on the **Time Level** pane. Plotting the Pb composition of the fluid on a log scale versus X position over the course of the simulation, we see that after 40 years of flushing with fresh water, Pb concentration remains well in excess of the drinking water standard of 15 µg/kg.

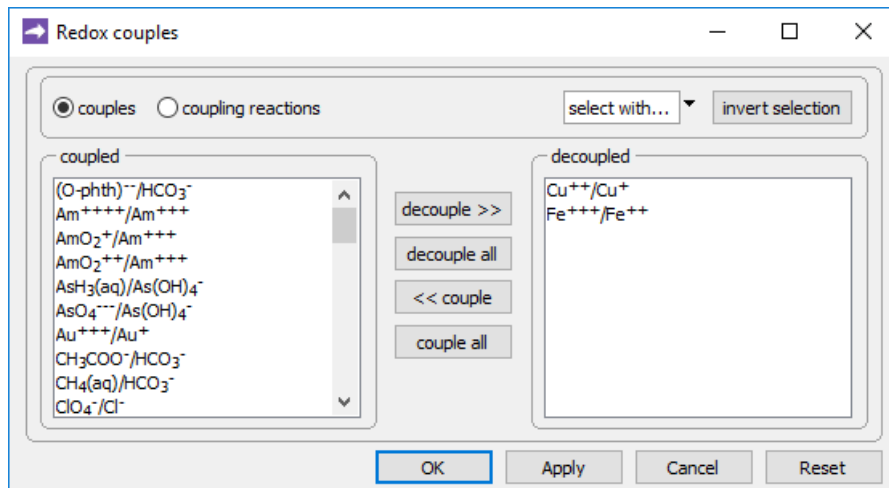


The slow decrease in Pb concentration during flushing reflects the gradual desorption of the metal from the aquifer sediment.

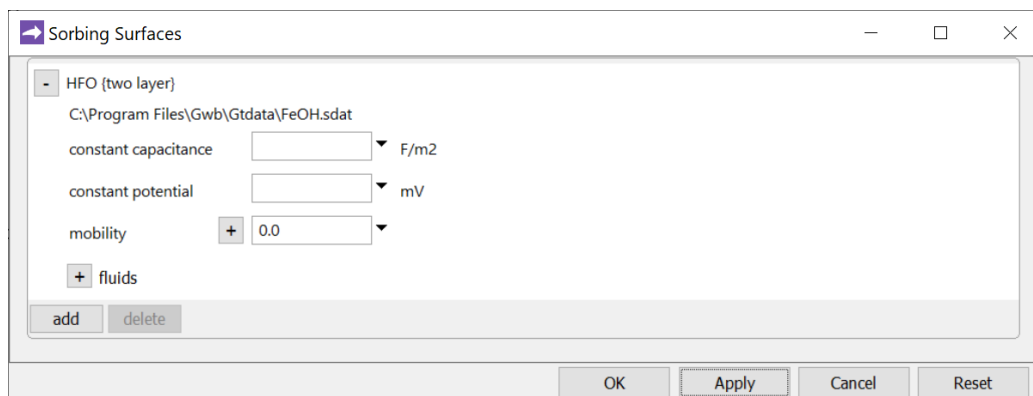
3.10 Example: groundwater chromatography

As a variation of the previous example, we consider the phenomenon of “groundwater chromatography.” In our calculation, we consider the migration of three divalent metal ions— Cu^{++} , Ni^{++} , and Pb^{++} —through the aquifer. The procedure is identical to the previous example, except for the addition of cupric copper and nickel. Double-click on the “Chromatography.x1t” input file to get started.

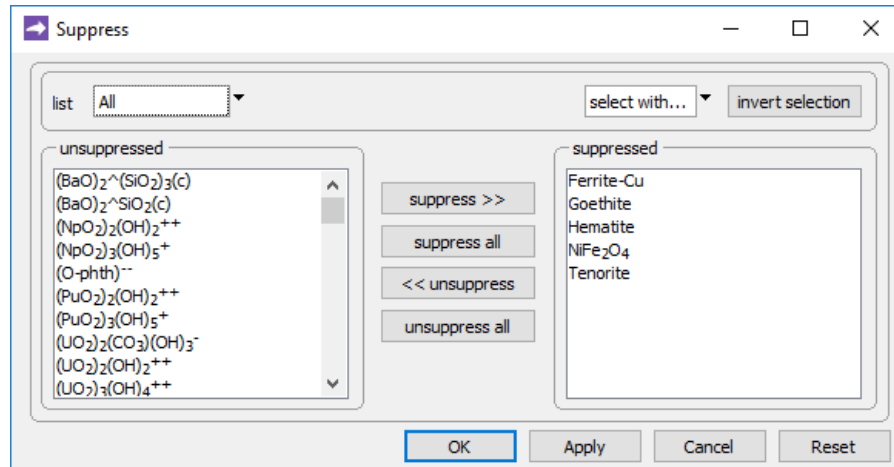
We use the two-layer model of surface complexation, as presented by Dzombak and Morel (1990), to model metal sorption and desorption. To invoke the model, we start by disabling the ferric-ferrous and cupric-cuprous redox couples from the **Config → Redox Couples...** dialog.



Next, we go to **File → Open → Sorbing Surfaces...** to read in the Dzombak and Morel reaction dataset.

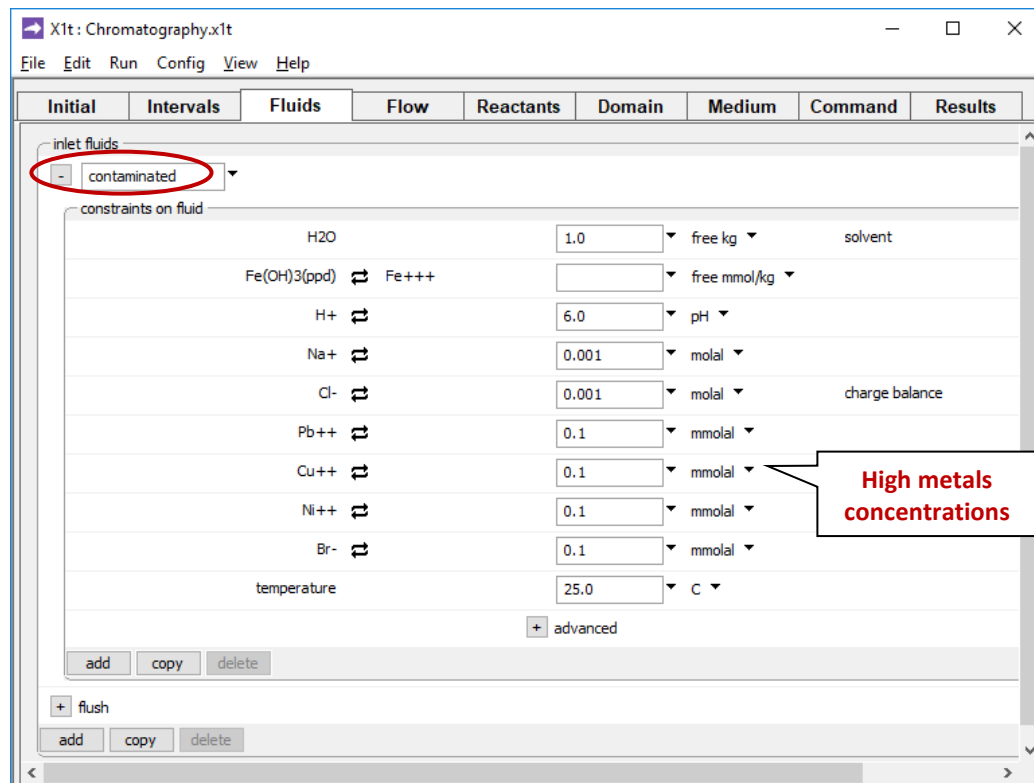


From the **Config** → **Suppress...** dialog,



we suppress the most stable ferric minerals to allow the iron to remain in ferric hydroxide form.

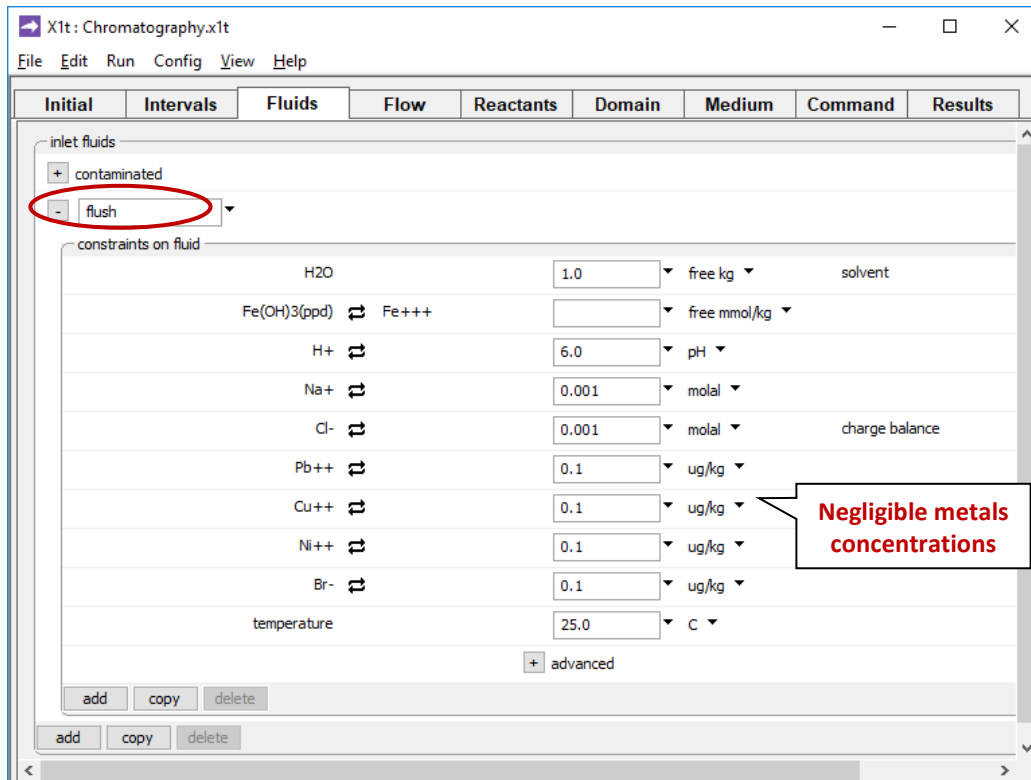
On the **Fluids** pane,



84 GWB Reactive Transport Modeling

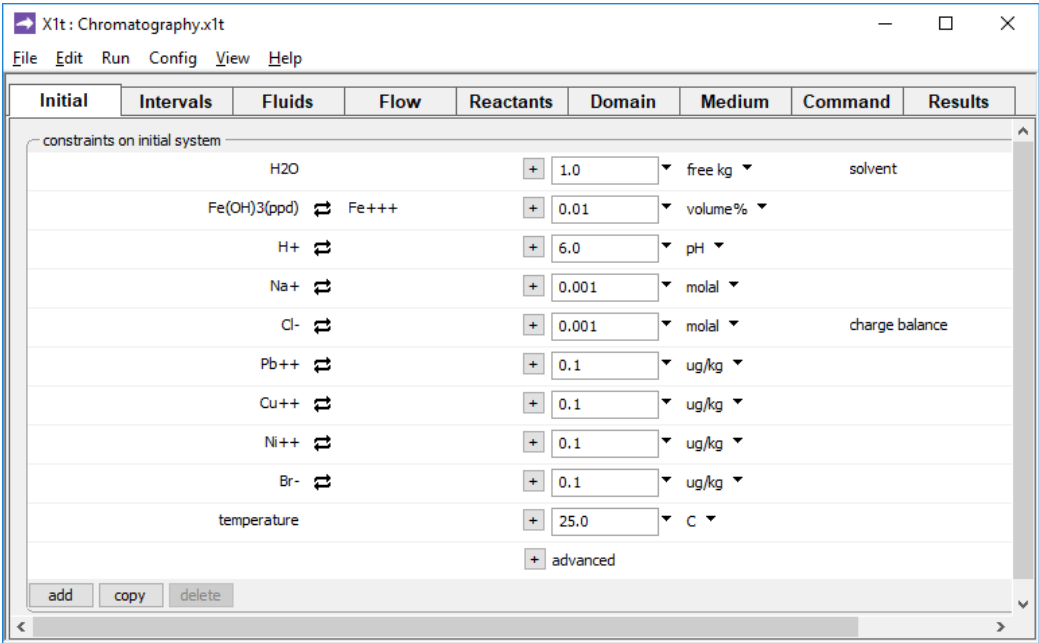
we set the contaminated fluid to a dilute water in equilibrium with ferric hydroxide [$\text{Fe}(\text{OH})_3$, a proxy for hydrous ferric oxide], containing 0.1 mmolal inorganic Pb^{++} , Cu^{++} , and Ni^{++} . The water also contains Br^- in equal molal concentration with the metals. Since, according to the surface complexation model, Br does not sorb, it serves in our calculation as a conservative tracer.

As before, we set the flushing fluid,



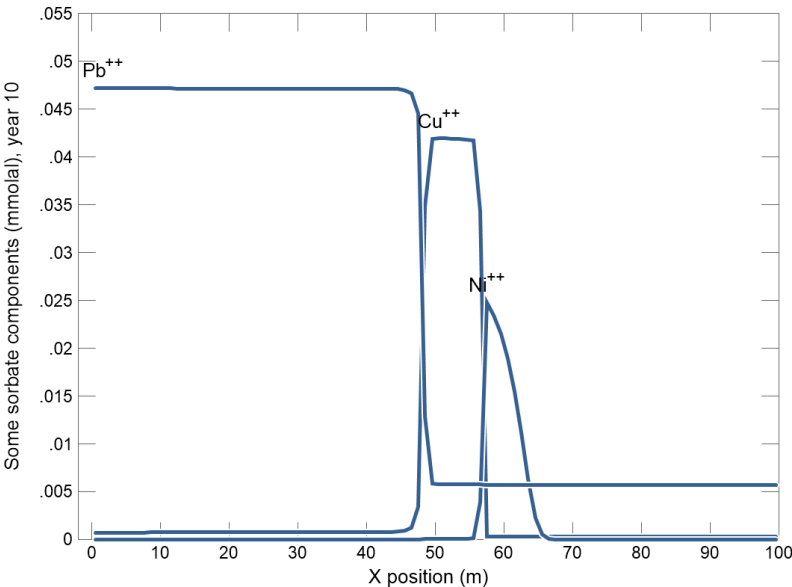
an uncontaminated water containing almost no heavy metals or bromide.

On the **Initial** pane,



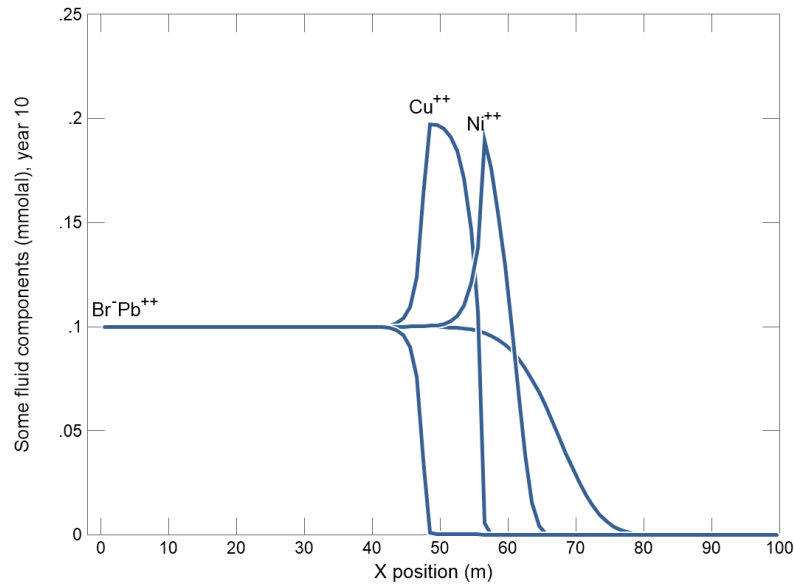
the aquifer, which includes a small amount of ferric hydroxide, contains an initially uncontaminated groundwater of the same composition as the flushing fluid.

Rendering the calculation results in **Xtplot**, we see that the sorbate forms a pattern of chromatographic banding emanating from the inlet.



Pb^{++} sorbs most strongly of the three metals, occupying the position nearest the inlet, followed by Cu^{++} and then Ni^{++} .

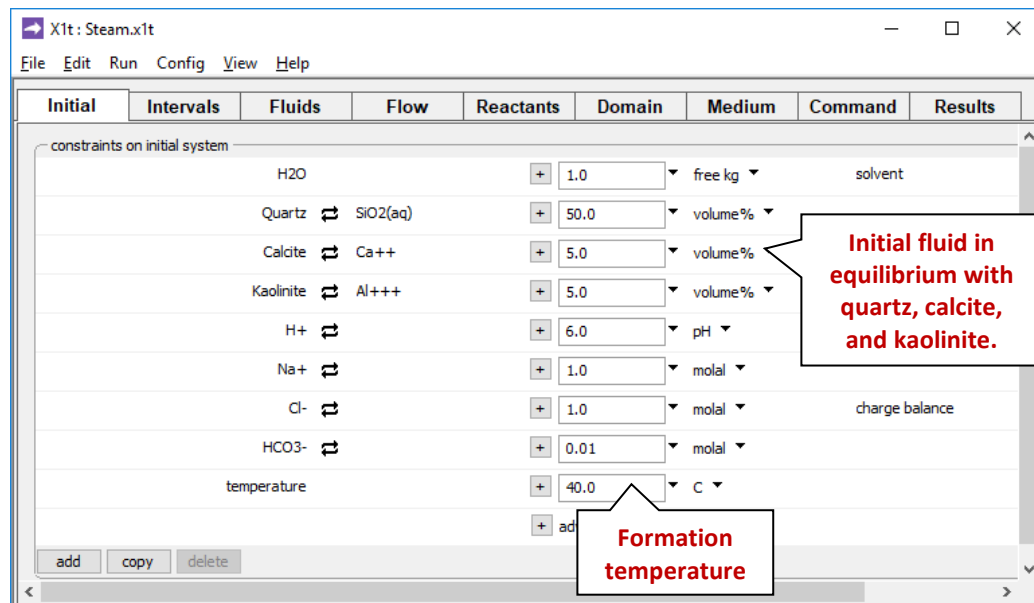
As the metal-bearing fluid invades the medium, Pb^{++} displaces Cu^{++} from the sorbing ferric hydroxide, and Cu^{++} displaces Ni^{++} , creating bands in which Cu^{++} and Ni^{++} concentrations in the medium exceed those in the inlet fluid.



3.11 Example: steam flood

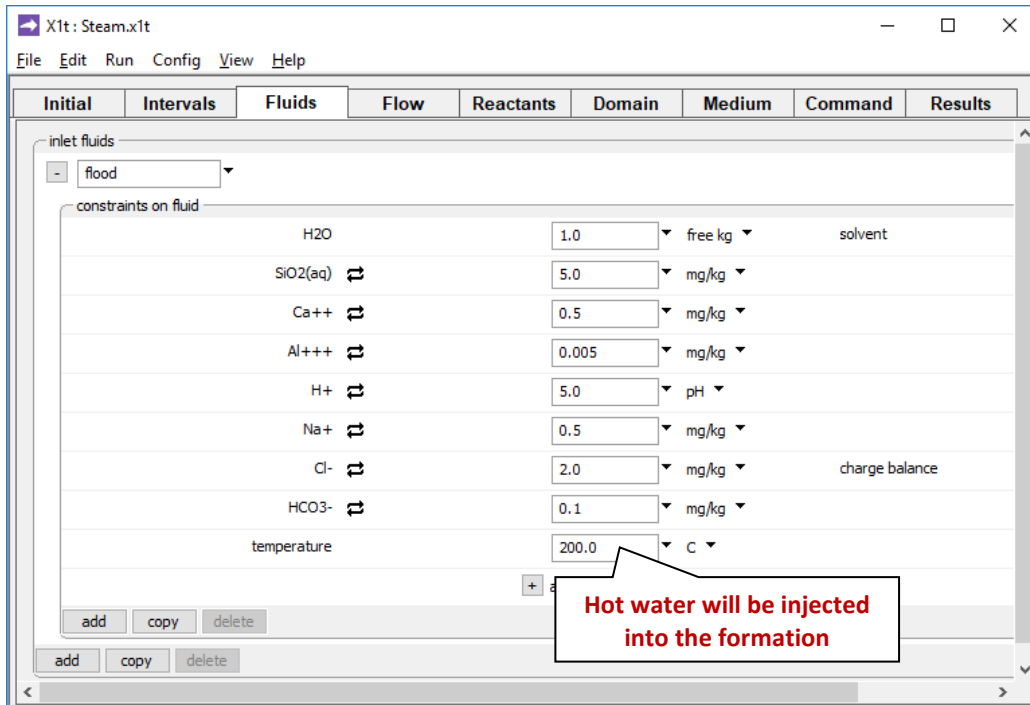
In this example, we will develop a polythermal model of transport and reaction in a clastic petroleum reservoir during steam flooding. Steam flooding is a procedure for enhanced recovery applied to reservoirs containing “heavy” oil. Steam flooding may be effective in increasing oil mobility, but the process can have undesirable effects when the injected fluid reacts chemically with minerals and fluids in the reservoir.

Start by double-clicking on the “Steam.x1t” input file. The **Initial** pane



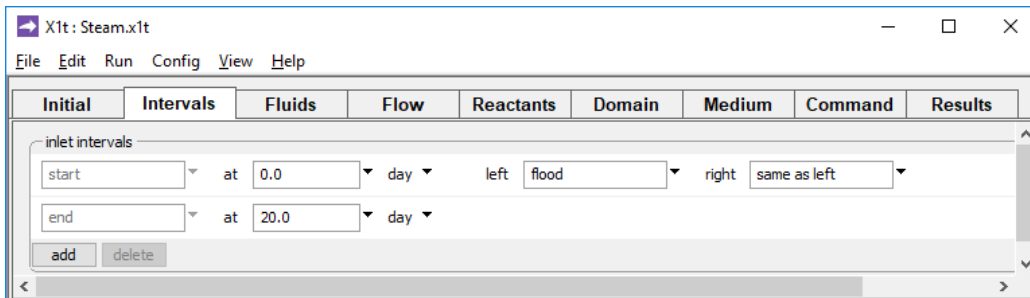
shows the conditions at the onset of the simulation. We assume that the unreacted formation contains 50% quartz, 5% calcite, and 5% kaolinite. These minerals constrain the $\text{SiO}_2(\text{aq})$, Ca^{++} , and Al^{+++} concentrations of the formation fluid. The fluid is a 1 molal NaCl solution containing a small amount of HCO_3^- ; its pH is 6. We set the initial temperature in the formation to 40°C.

On the **Fluids** pane,



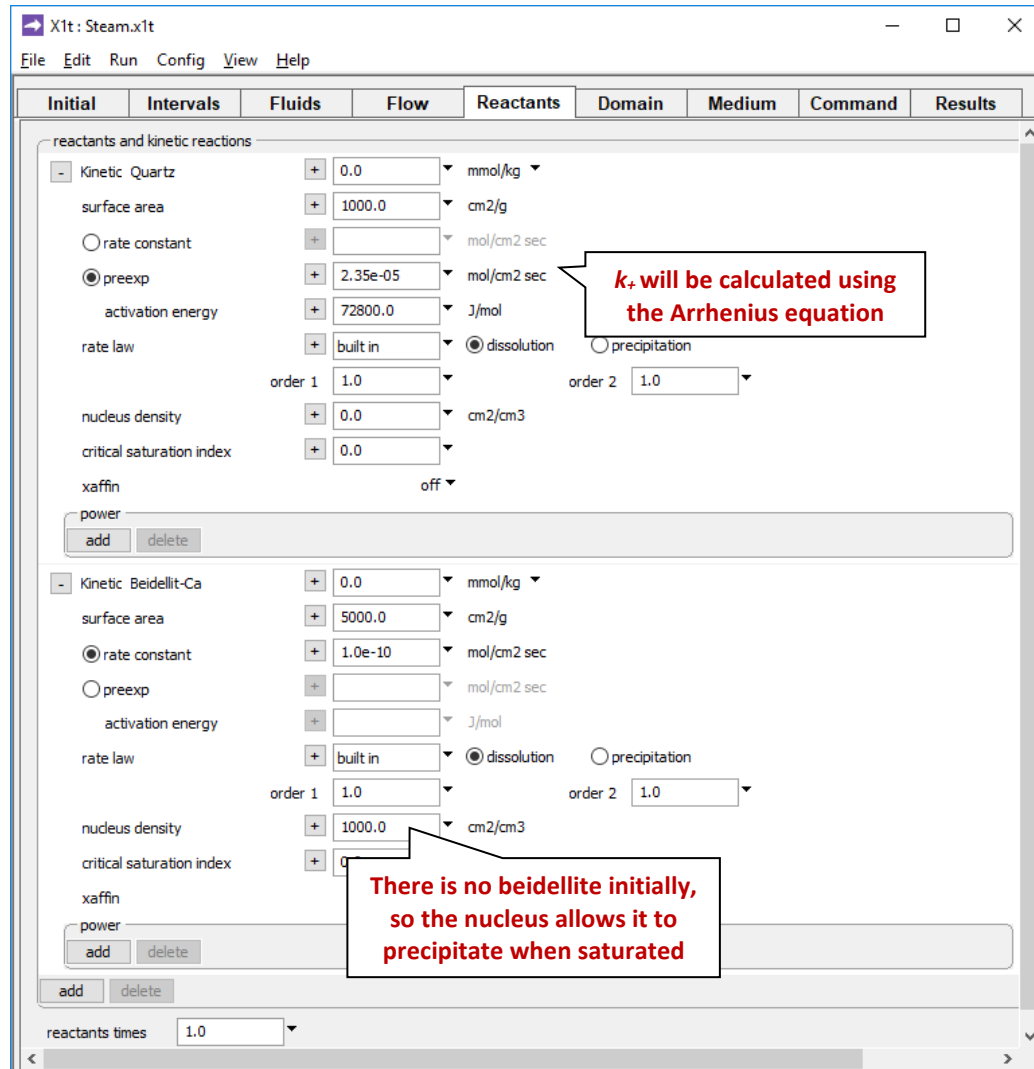
we set a boundary fluid “flood” to represent the steam (actually hot water). The fluid’s composition reflects fresh water heated to 200°C.

On the **Intervals** pane, we associate the “flood” with the domain’s left side,



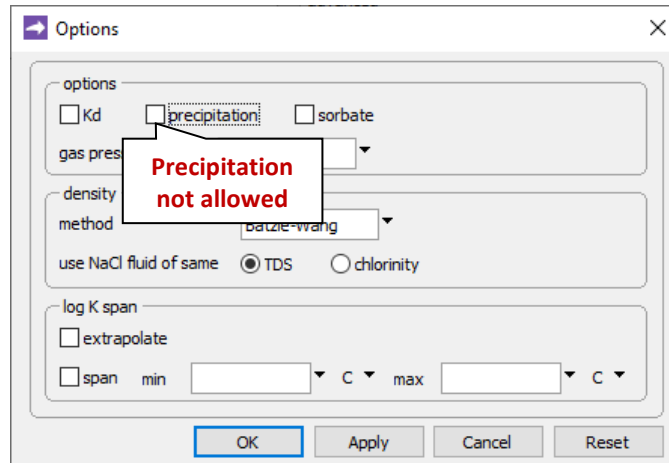
and let the simulation run for 20 days.

On the **Reactants** pane, we assign rate laws for quartz and calcium beidellite.



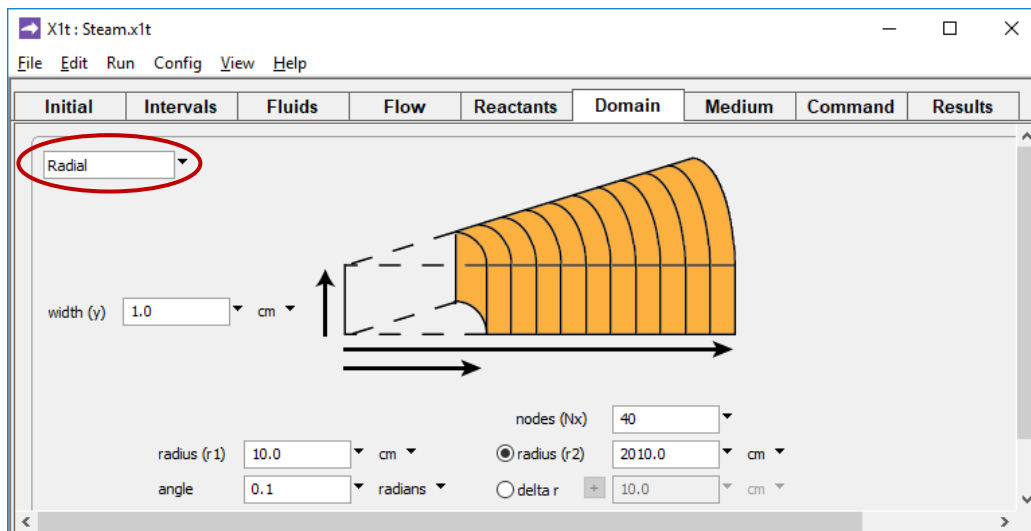
Since the simulation is polythermal, we set the rate constant k_+ for quartz in terms of an activation energy E_A and a pre-exponential factor A . Finally, we set a kinetic law for calcium beidellite fast enough to allow the mineral to precipitate and remain in equilibrium with the evolving fluid, once it becomes saturated.

Going to **Config** → **Options...**,



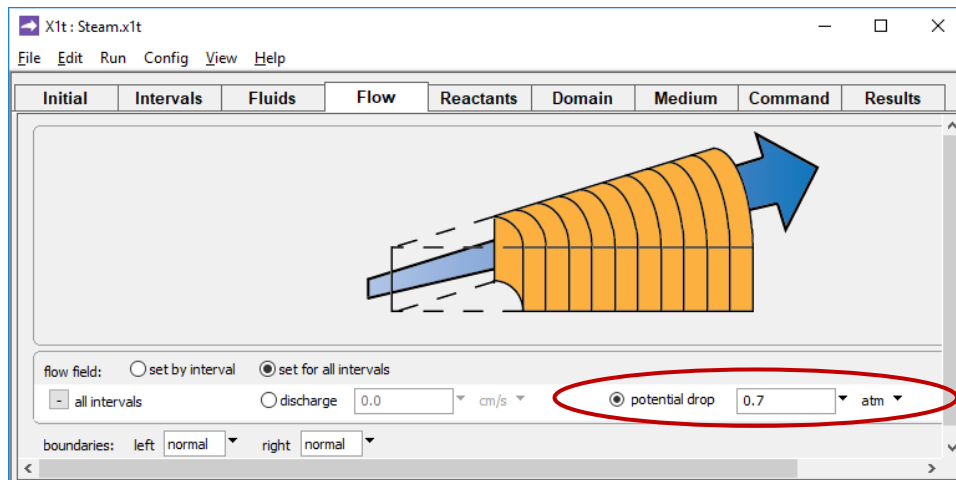
the precipitation option has been deselected. As a result, no minerals are allowed to form except for those in the initial system and those for which we set kinetic rate laws.

On the **Domain** pane, we configure the model in radial coordinates.



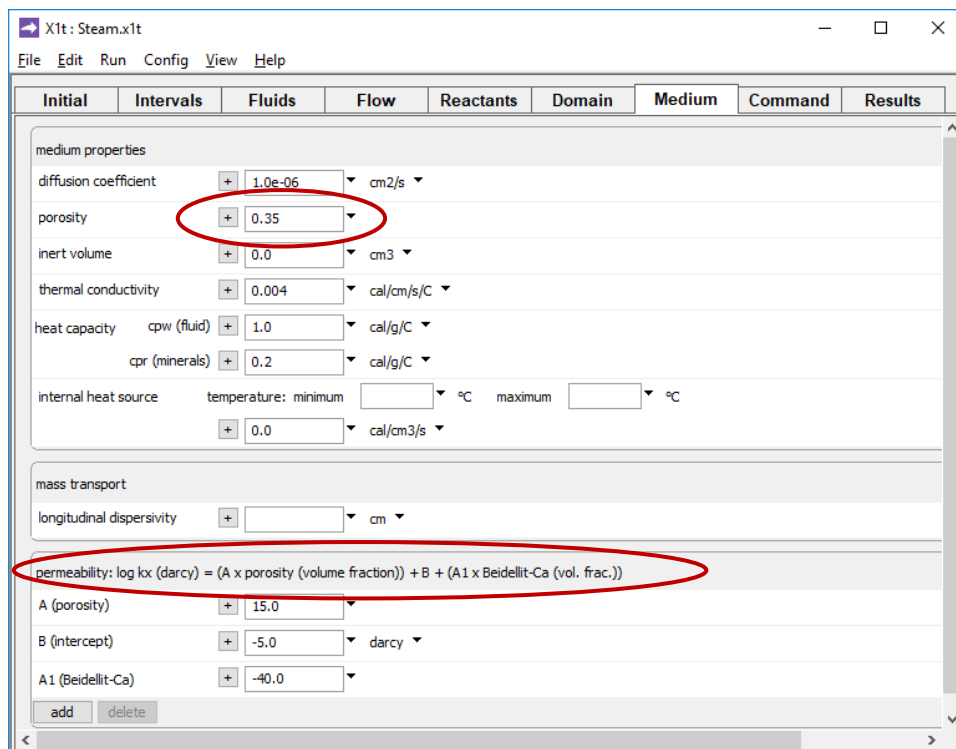
The domain starts at a radius of 10 cm and continues outward for 2000 cm.

On the **Flow** pane,



we assume an injection pressure 0.7 atm greater than the formation pressure to provide a drive for flow.

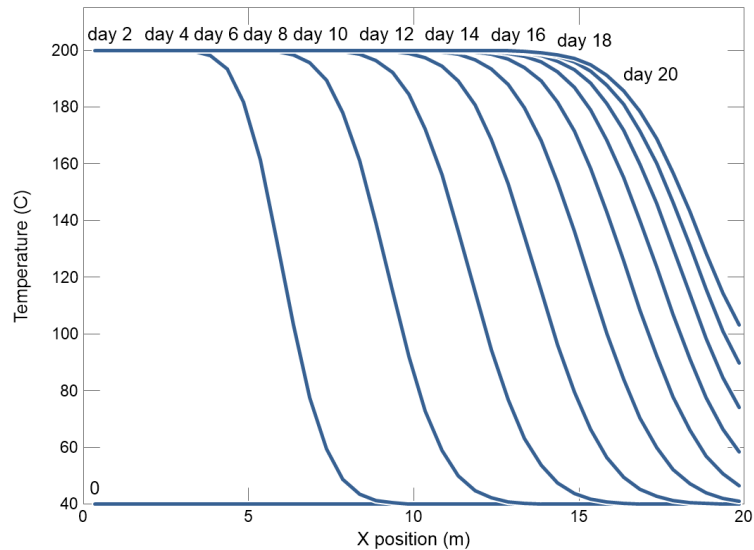
On the **Medium** pane,



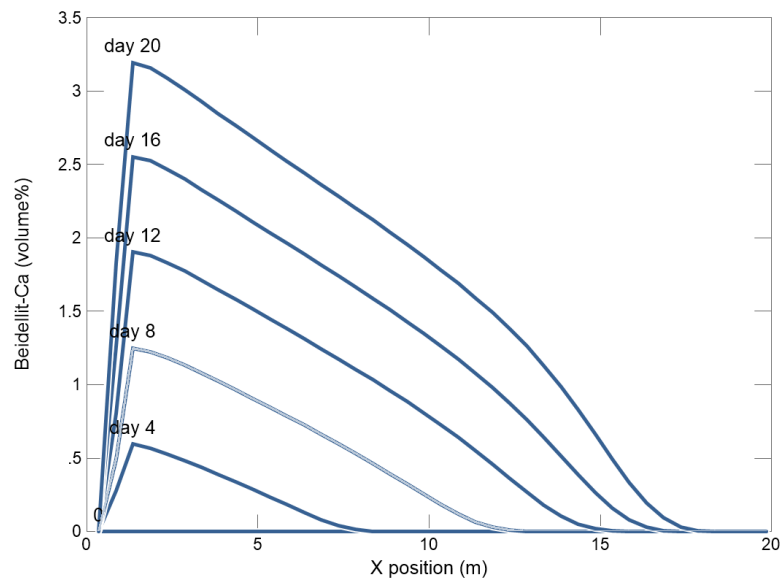
92 *GWB Reactive Transport Modeling*

we set the initial porosity and define a correlation by which permeability varies with porosity and the volume fraction of calcium beidellite, a smectite mineral.

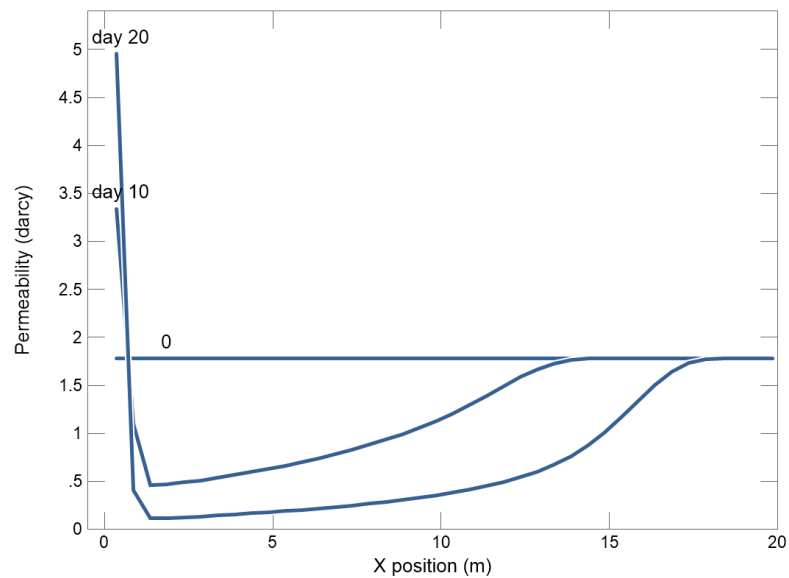
Select **Run** → **Go** to calculate the model. In the simulation results, we see that a front of hot water moves radially outward from the well.



As temperature within the formation increases, the calcite, quartz, and kaolinite react to form calcium beidellite and CO_2 . With time, the beidellite comes to occupy a significant fraction of the formation's volume.



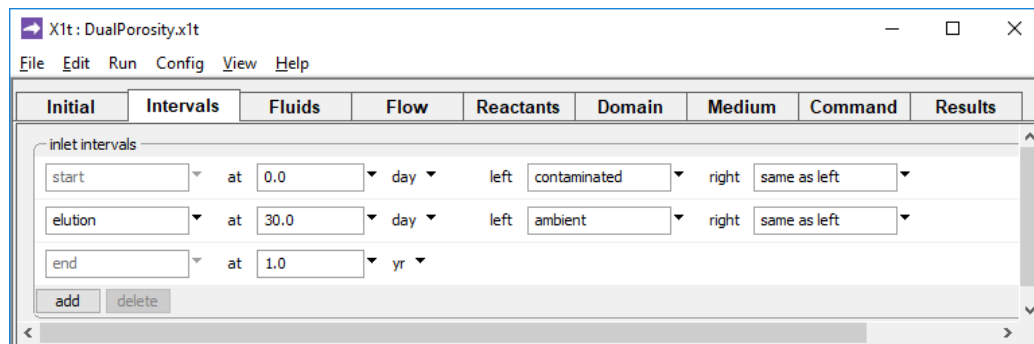
The clay formed by this reaction serves to decrease permeability, hence damaging the formation.



3.12 Example: dual porosity model

We consider how a pulse of Pb^{++} -contaminated water might migrate through an aquifer, part of which is stagnant. Contaminated fluid in the simulation enters the domain for a month, after which time the aquifer is flushed with clean water. Throughout the simulation, Pb^{++} can diffuse from the flowing groundwater into the stagnant zones, and back out.

Start by double-clicking on the “DualPorosity.x1t” input file. On the **Intervals** pane,



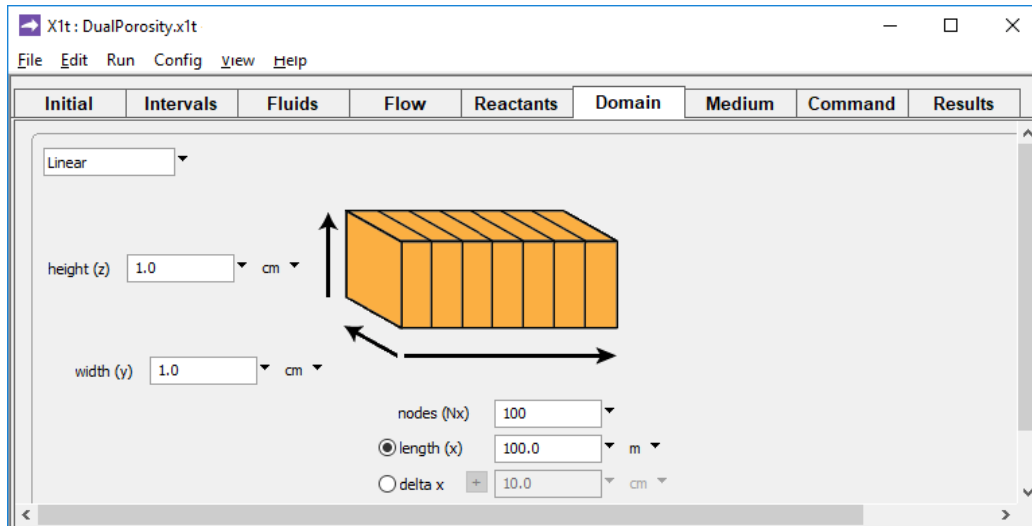
we set an **X1t** simulation composed of two intervals, the first spanning 1 month, and the second extending from that time to 1 year.

Going to **Config → Output...**,

we set the program to write results in 30 day increments, and on **Config → Stepping...**

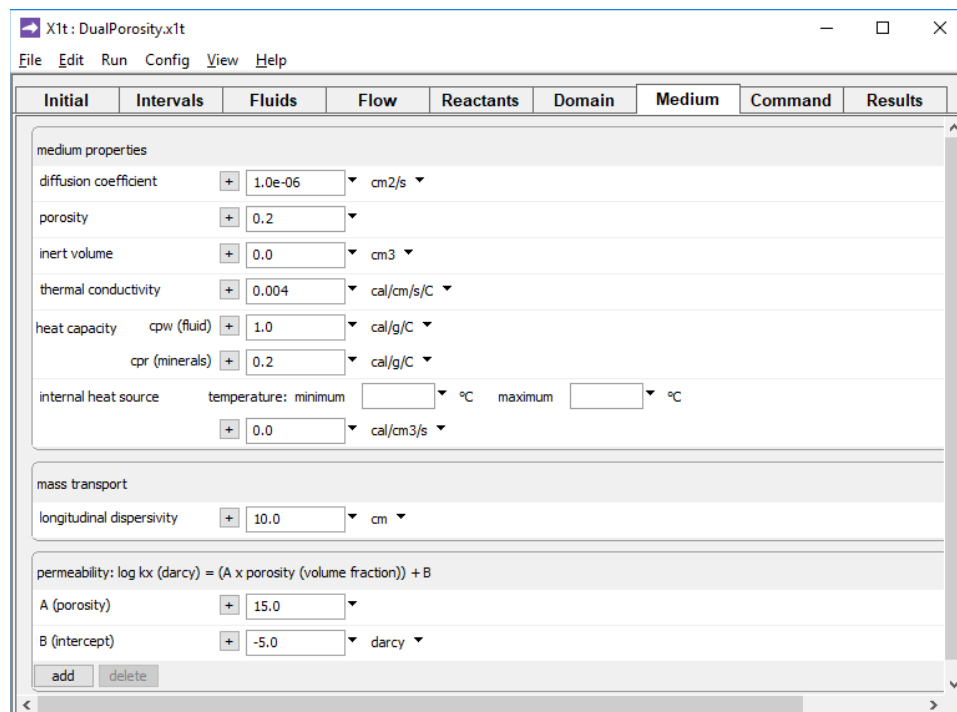
we specify a starting step “dx_init” for the time marching of 10^{-3} .

On the **Domain** pane,



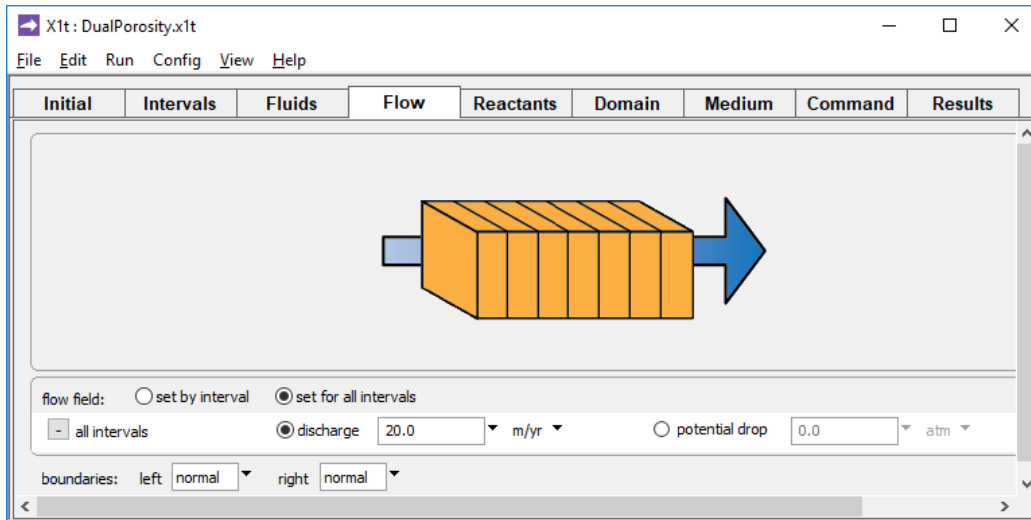
we take the aquifer to be 100 m long, composed of 1 m long nodal blocks.

On the **Medium** pane,



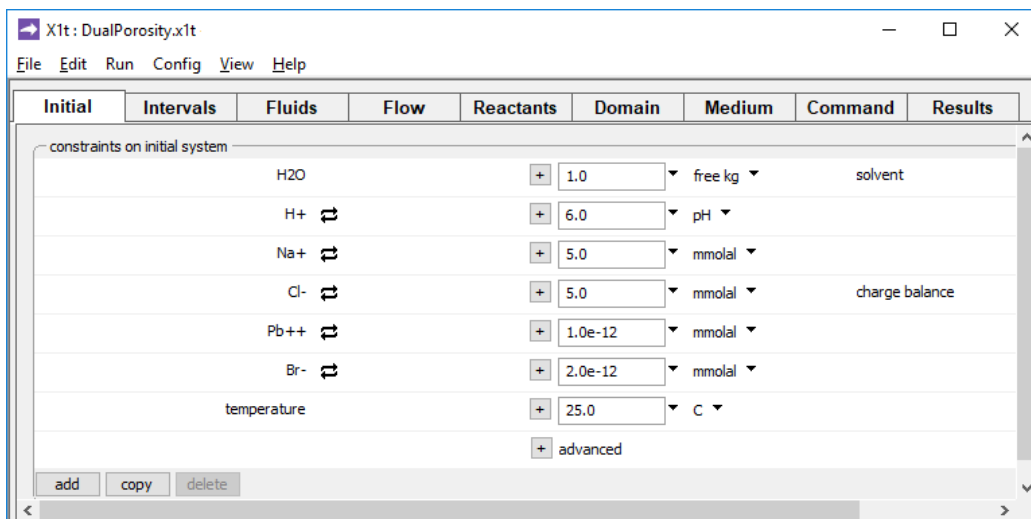
we set a porosity of 20% and a dispersivity of 10 cm.

From the **Flow** pane,



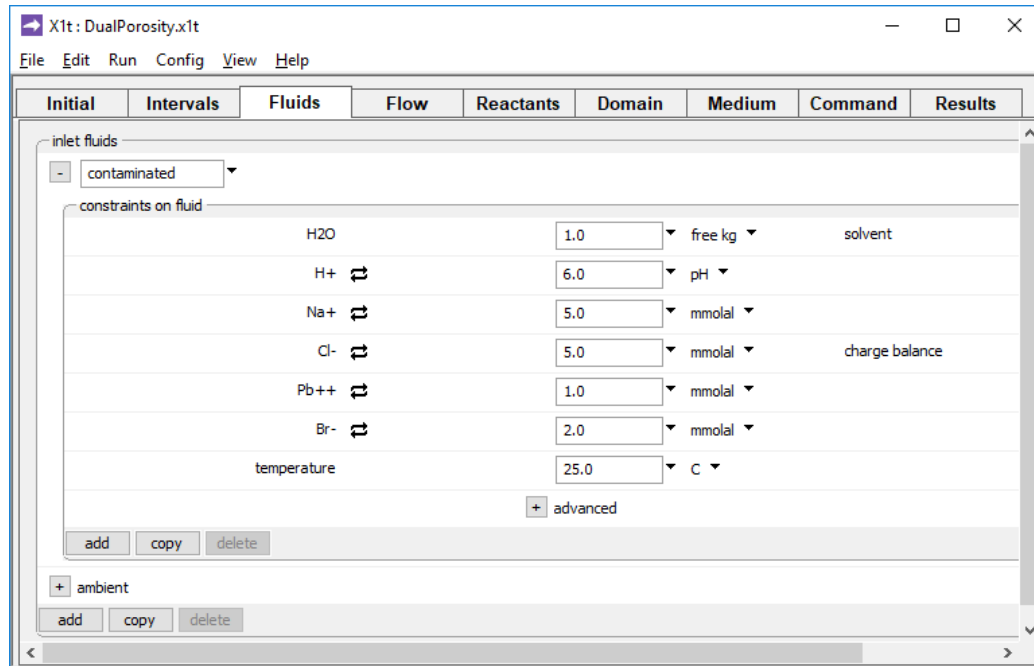
we specify that water recharges the left side of the domain at a specific discharge of 20 m/yr.

On the **Initial** pane,



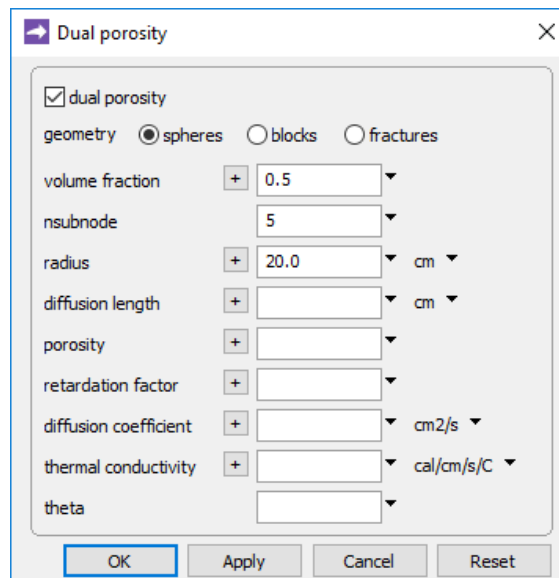
the initial fluid is a NaCl solution of pH 6 containing almost no Pb⁺⁺.

On the **Fluids** pane,



the inlet fluids for the two intervals in the simulation are the same as the initial fluid, except the fluid for the first interval contains 1 mmolal Pb⁺⁺, with Br as the counterion.

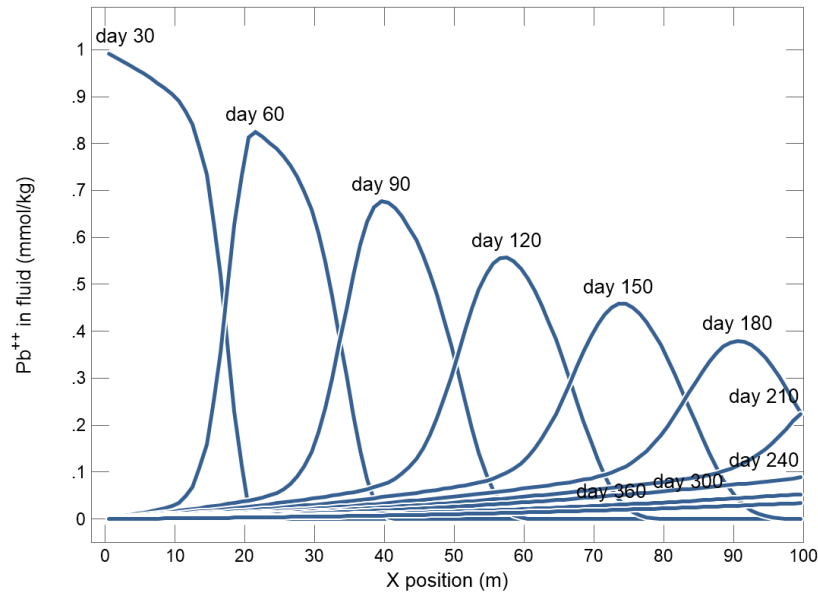
To account for the effects of the stagnant fraction of the aquifer, we define a dual porosity model and run the simulation.



98 *GWB Reactive Transport Modeling*

The stagnant zones in the model are composed of spheres 40 cm across that make up half the aquifer's volume.

Run → Go calculates the model. The model results show a pulse of lead migrating through the aquifer.



We can compare this result to the case in which the Pb^{++} does not diffuse into the stagnant zone by setting the diffusion coefficient there to 0.

Dual porosity

☒ dual porosity

geometry ☒ spheres ☐ blocks ☐ fractures

volume fraction + 0.5

nsubnode 5

radius + 20.0 cm

diffusion length + cm

porosity +

retardation factor +

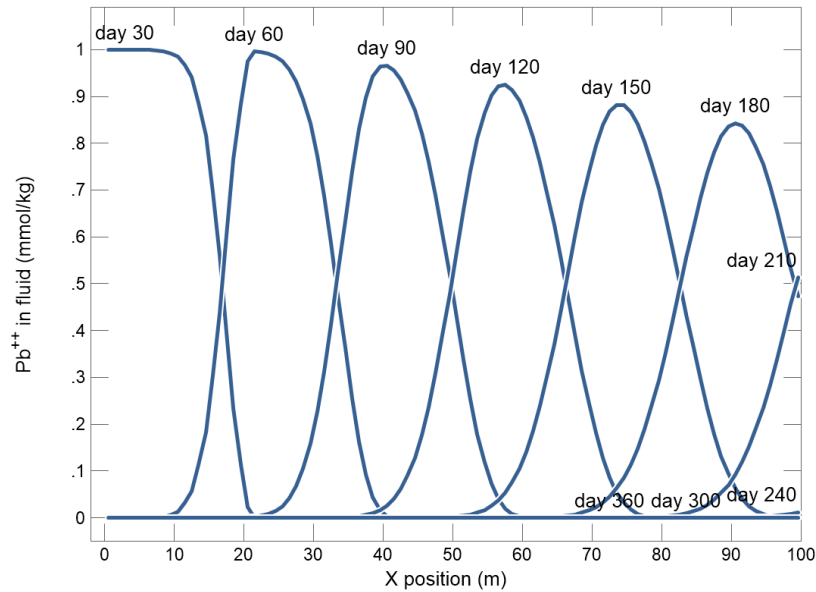
diffusion coefficient + 0 cm²/s

thermal conductivity + cal/cm/s/C

theta

OK Apply Cancel Reset

Re-running the simulation gives the result:

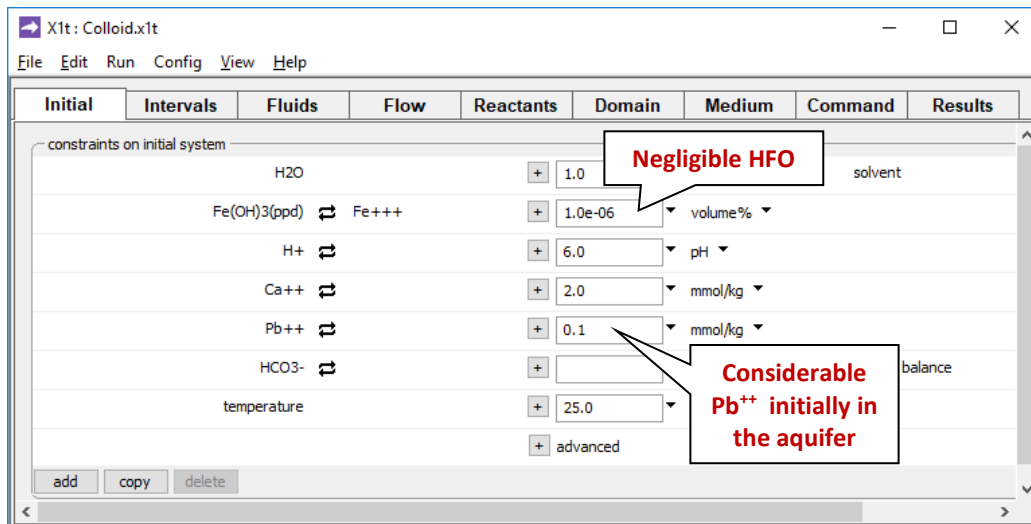


Comparing the plots, we see the effect of the dual porosity model is to attenuate the concentration of Pb^{++} as it migrates, and retard its displacement from the aquifer.

3.13 Example: mobile colloid

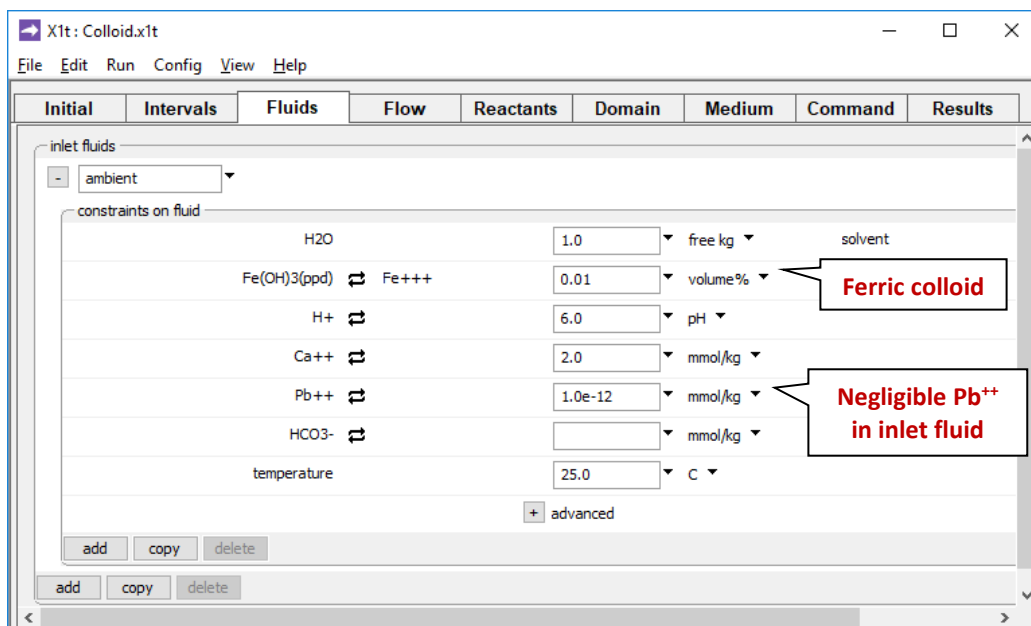
In this example, we model how a ferric colloid migrating with groundwater along an aquifer might displace Pb^{++} from the aquifer sediments. Initially, inorganic lead is almost entirely sorbed to the sediment grains, leaving little in solution. With time, a fluid carrying a ferric hydroxide colloid, which complexes strongly with Pb^{++} , passes into the aquifer.

To start, double-click on file “Colloid.x1t”. Moving to the **Initial** pane,



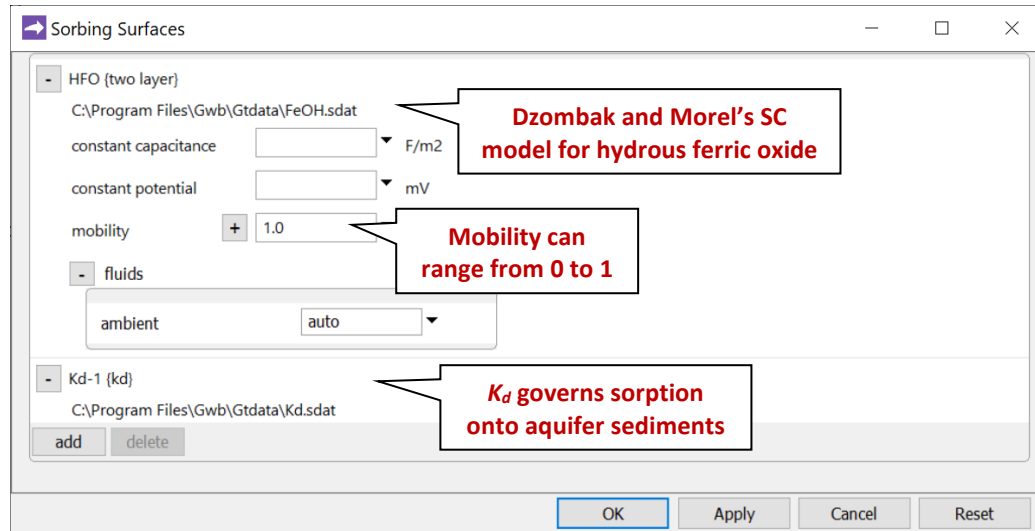
we see the aquifer contains a negligible amount of the ferric precipitate $\text{Fe}(\text{OH})_3$, a Ca-HCO_3 water, and a considerable amount of inorganic lead distributed between solution and sorbate.

On the **Fluids** pane,



fluid passing into the domain is the same as the initial fluid, except it contains more abundant ferric precipitate, but almost no Pb^{++} .

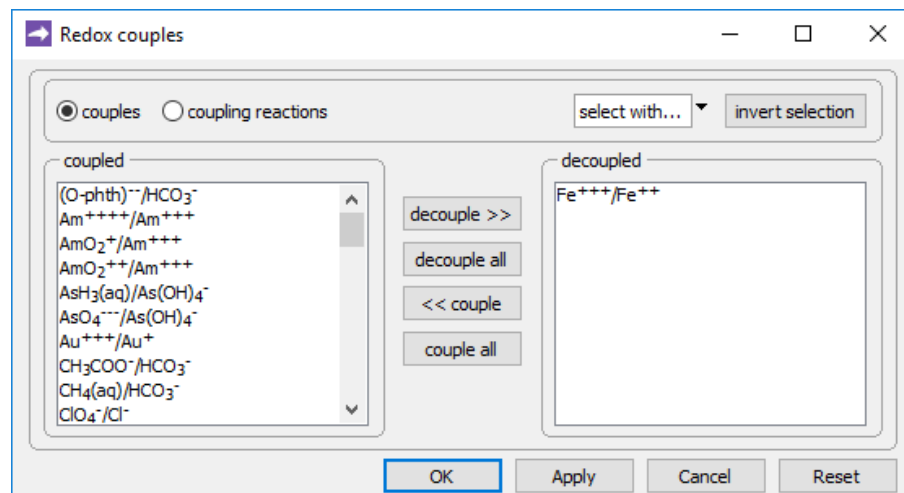
Opening **Config → Sorbing Surfaces...**, we see we've read in the "Kd.sdat" and "FeOH.sdat" datasets.



The former supplies a distribution coefficient K_d governing sorption of Pb^{++} onto the aquifer sediments. The latter contains Dzombak and Morel's (1990) compilation of surface complexation reactions for hydrous ferric oxide.

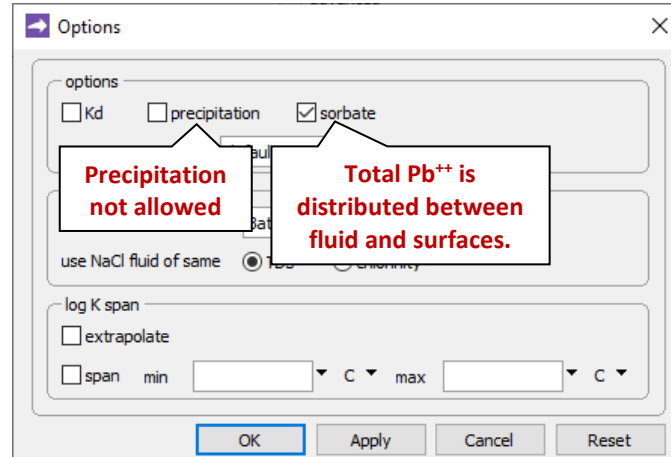
We'll set the mobility of the ferric colloid in the domain to 1 in order to make the $\text{Fe}(\text{OH})_3$ precipitate and its sorbate into a mobile colloid. Its mobility in the inlet fluid is by default set equal to the first interior node in the domain.

On the **Config → Redox Couples...** dialog,



we see the redox coupling between ferric and ferrous iron has been disabled. This setting explains why we've been able to include Fe^{+++} species and minerals in the simulation, without setting basis entries representing Fe^{++} or redox state.

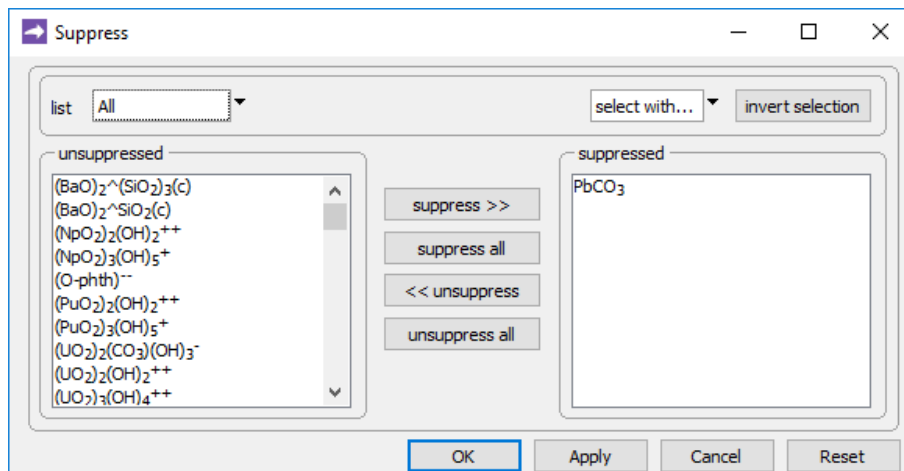
Going to **Config → Options...**,



the precipitation option has been deselected. This setting prevents stable but slow-forming ferric minerals like goethite (FeOOH) from appearing in the simulation, at the expense of the $\text{Fe}(\text{OH})_3$ precipitate.

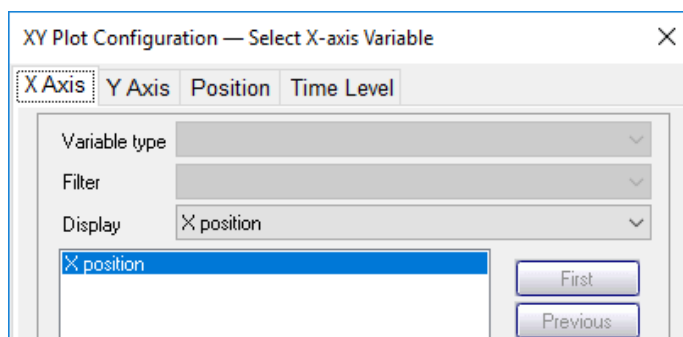
We also see that the “sorbate” option has been selected. As a result, the total Pb^{++} concentration we specify on the **Initial** and **Fluids** panes is distributed between the fluid and the sorbate.

On the **Config → Suppress...** dialog,

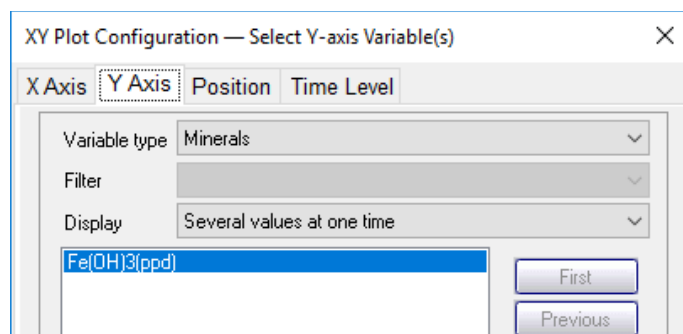


we've suppressed the PbCO_3 ion pair, the stability of which, in the database, is almost certainly erroneous.

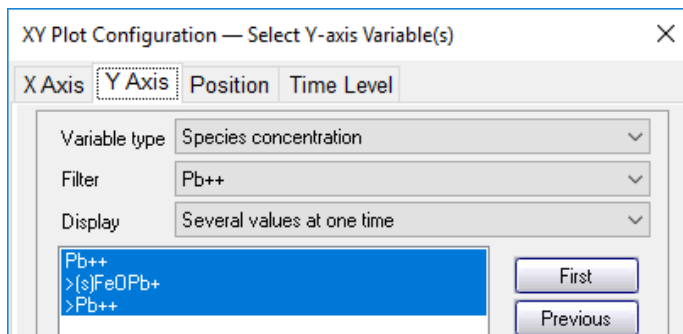
Trace the simulation by selecting **Run → Go**. Use **Xtplot** to make a graph showing the mass of the sorbing mineral $\text{Fe}(\text{OH})_3$ throughout the domain. Double-click on the x-axis label, and for “Display”, choose “X position”.



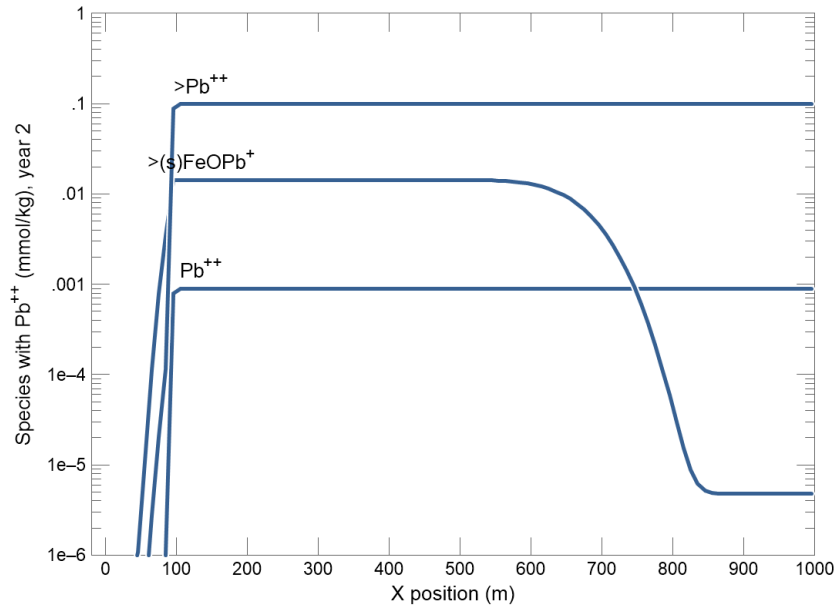
On the **Y Axis** pane, choose “Minerals” and then select “ $\text{Fe}(\text{OH})_3(\text{ppd})$ ”.



Now plot the concentrations of the various Pb^{++} species along the aquifer. On the **Y Axis** pane, choose “Species concentration”, and for “Filter”, select “ Pb^{++} ”. Then select “ Pb^{++} ”, “ >Pb^{++} ”, and “ >(s)FeOPb^{++} ”.



In the calculation results, shown here two years into the simulation,

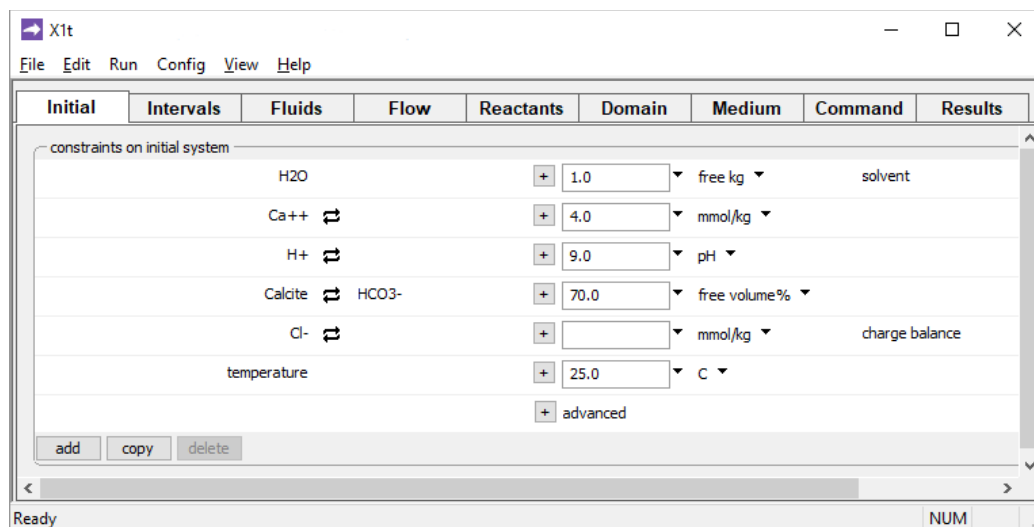


we see how the mobile colloid scavenges sorbed inorganic lead Pb^{++} from solid grains in the aquifer, complexes it to the colloid surface as >(s)FeOPb^+ , and mobilizes it along the direction of groundwater flow.

3.14 Example: stable isotope transport

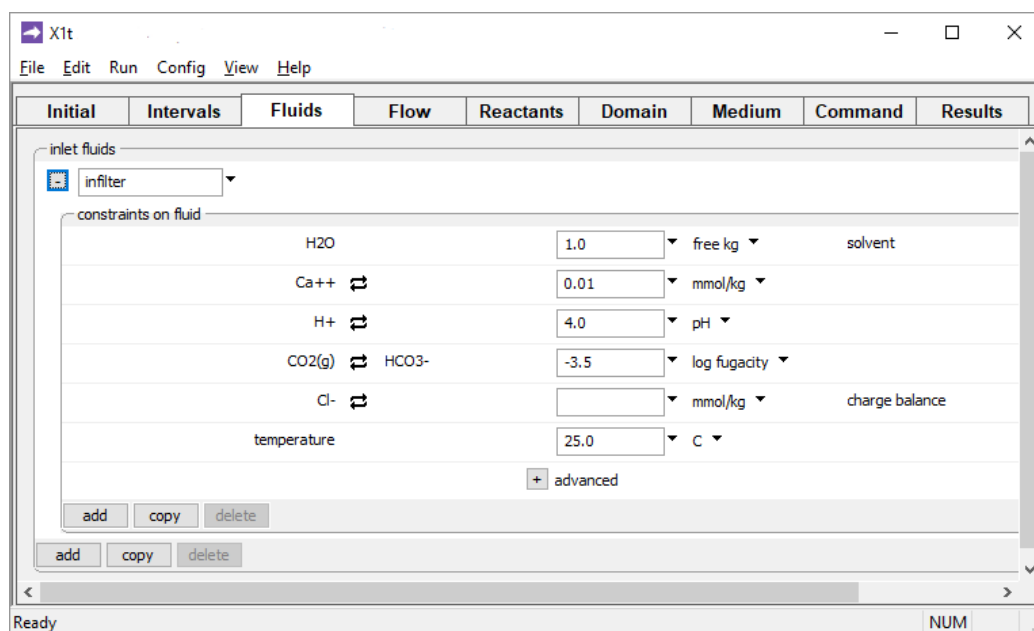
In this example, we trace fractionation of the carbon isotope ^{13}C as a dilute water in equilibrium with CO_2 in the atmosphere infiltrates a limestone holding a more concentrated, isotopically heavier water. When the dilute, slightly acidic water moves into the limestone, calcite dissolves into it according to a kinetic rate law. The reaction consumes H^+ from the water, driving it to more alkaline pH, and adds Ca^{++} and ^{13}C -rich carbonate, making it more concentrated and enriching it in ^{13}C .

Start by double-clicking on "IsotopeTransport.x1t". When **X1t** opens, move to the **Initial** pane



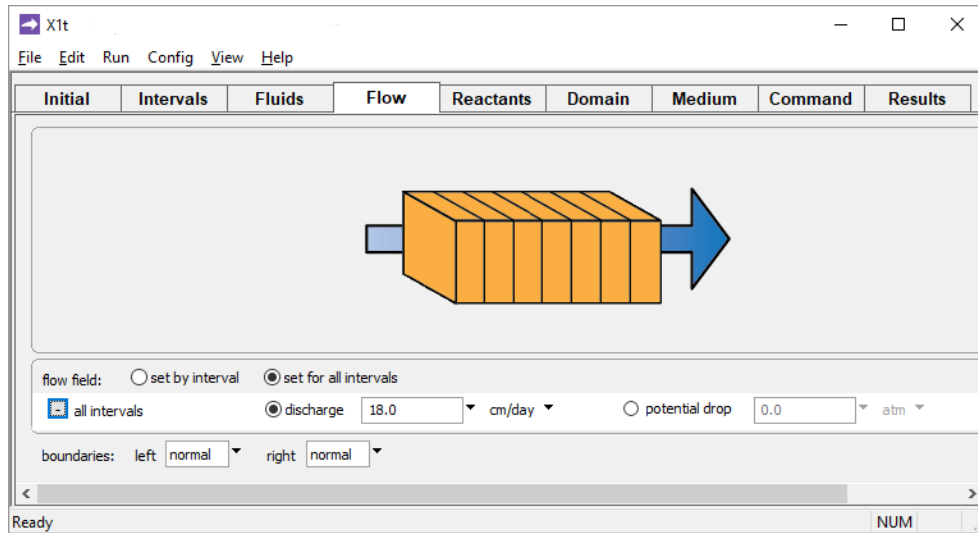
Here we see that calcite makes up 70% of the initial system's volume, leaving a porosity of 30%, and that equilibrium with the mineral constrains bicarbonate concentration. The pH is set to 9 and the initial fluid contains 4 mmol/kg Ca^{++} , as well as chloride to balance charge.

The **Fluids** pane

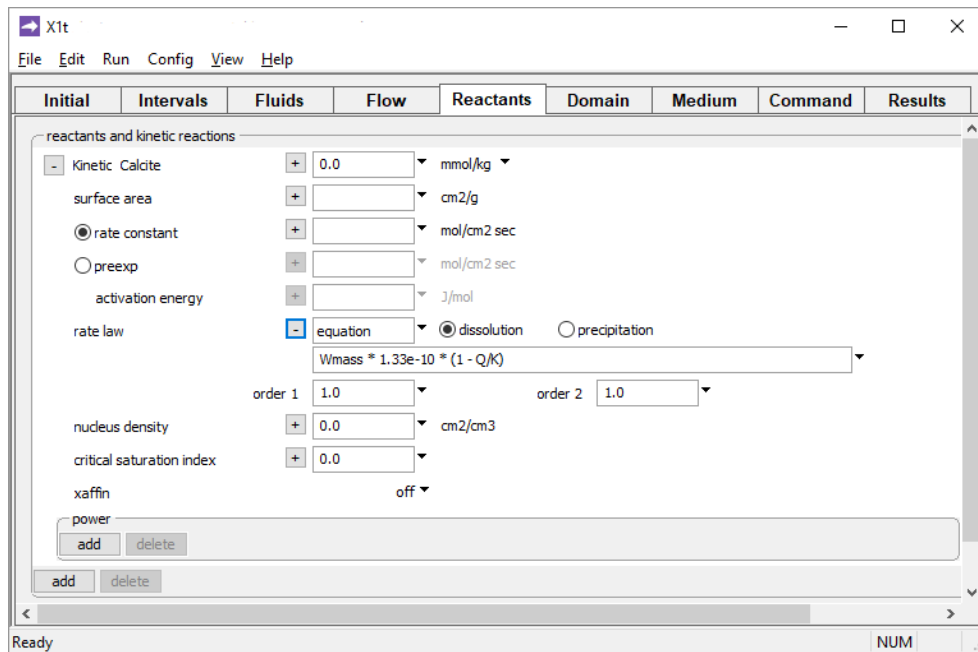


defines a water “infiltrer” that is more acidic and dilute than the initial fluid, and is in equilibrium with atmospheric CO_2 .

On the **Flow** pane,

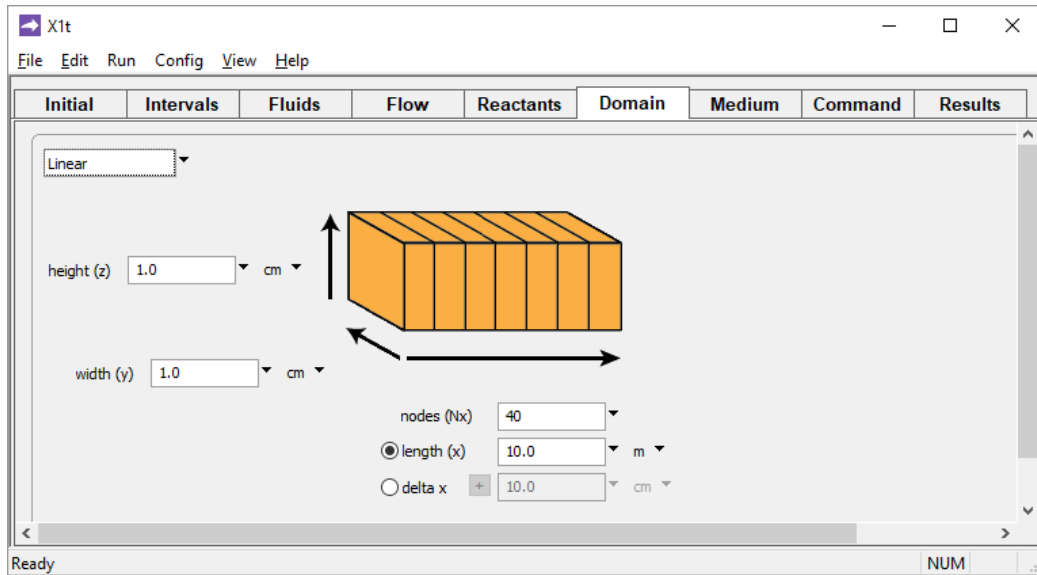


we set discharge of the infiltrating fluid into the domain to 18 cm/day. Then, on the **Reactants** pane

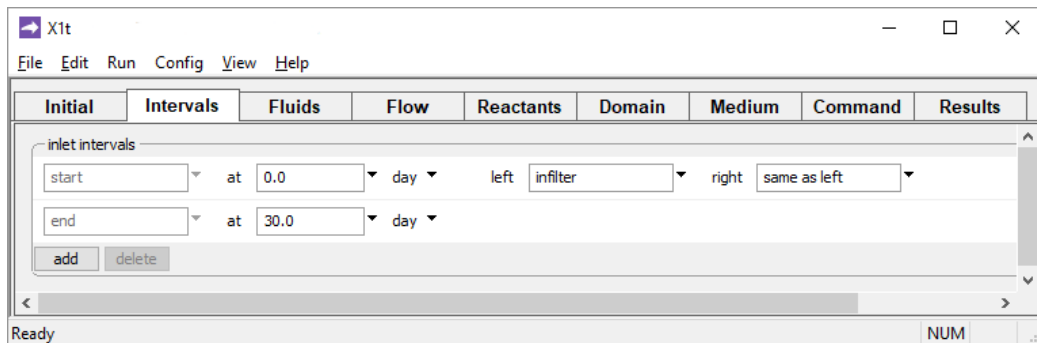


we define a simple kinetic rate law that describes dissolution of calcite into the infiltrating water, as a function of water mass and the mineral's saturation Q/K .

On the **Domain** pane, we see the system is 10 m long and divided into 40 nodal blocks



Moving to the **Intervals** pane,



time span in the model is set to 30 days.

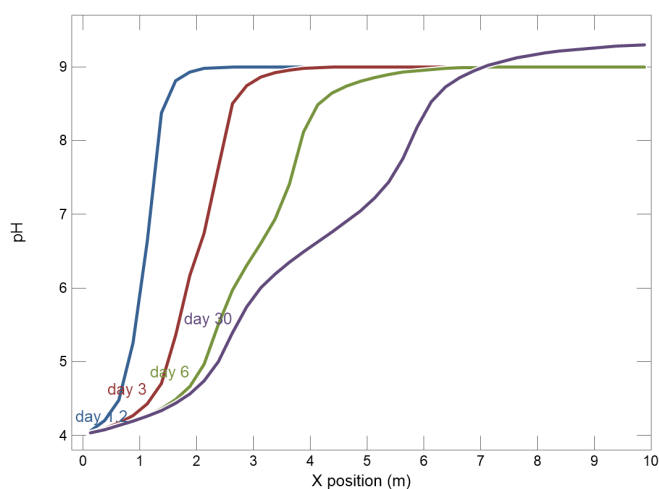
Finally, on the **Config** → **Isotopes...** dialog, we see calcite in the limestone is segregated from isotopic exchange, as is appropriate at low temperature, and its isotopic composition $\delta^{13}\text{C}$ is set to +1‰ relative to the standard we choose to carry in the calculation, which is PDB. The initial fluid has a composition of –5‰ and the infiltrating fluid, –22‰. Since the input is relative to the PDB standard, the model will report results on this scale.

The screenshot shows the 'Isotopes' dialog box with the following settings:

- segregated minerals:**
 - Calcite 100.0%
- isotopic composition:**
 - initial:**
 - Hydrogen-2: [] ‰
 - Carbon-13: -5.0 ‰
 - Oxygen-18: [] ‰
 - Calcite: 1.0 ‰
 - infiltrator:**
 - Hydrogen-2: [] ‰
 - Carbon-13: -22.0 ‰
 - Oxygen-18: [] ‰
 - Sulfur-34: [] ‰

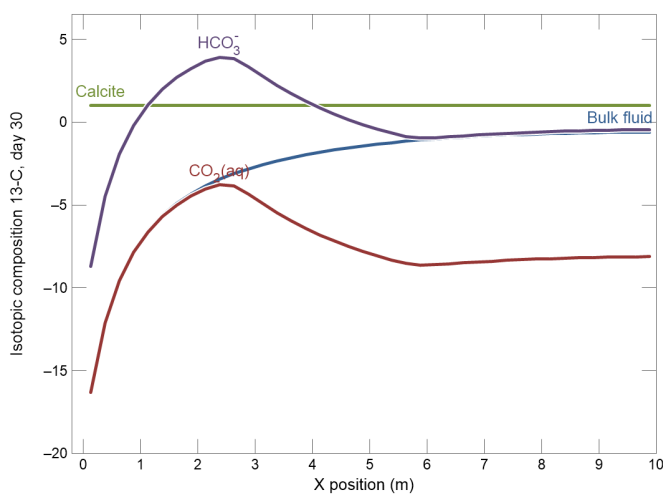
Trigger the calculation by selecting **Run** → **Go**. Move to the **Results** pane and click on **Plot Results** to graph how pH as a function of position in aquifer changes during the infiltration.

As weakly acidic fluid enters the domain, pH of the aquifer decreases toward the inlet and, in response, calcite starts to dissolve, consuming acid



After about 20 days, the pH profile across the domain converges to a steady state in which supply of H^+ by the infiltrating water and consumption of acid by calcite dissolution fall into balance.

The isotopic compositions of individual species and bulk phases also reach a steady state by the end of the simulation



Calcite is unchanged from the input composition, because we have segregated the mineral from isotope exchange. The bulk fluid becomes progressively heavier as it moves through the domain, reflecting the addition of ^{13}C from the dissolving calcite.

Near the inlet in the simulation, pH is low enough that $\text{CO}_2(\text{aq})$ is the dominant carbonate species, and hence the composition of the bulk fluid traces that species' trajectory. Past about 2 m into the domain, bicarbonate becomes progressively more important in the carbonate speciation as pH moves higher. In response, $\delta^{13}\text{C}$ for the bulk fluid deviates from the composition of $\text{CO}_2(\text{aq})$ toward that of the bicarbonate species.

Once the fluid has traversed the domain, its bulk $\delta^{13}\text{C}$ is about -0.6‰ . This composition reflects the fact that the reacted fluid contains around .01 mmol/kg of atmospheric carbon ($\delta^{13}\text{C} = -22\text{‰}$) and about .14 mmol/kg of carbon supplied by the dissolution of calcite ($\delta^{13}\text{C} = +1\text{‰}$). The bulk composition is the weighted average of the two carbon sources.

3.15 X1t command line

You can start **X1t** by clicking the icon on the GWB dashboard, opening an “.x1t” file, or entering the command `x1t.exe` from the Windows “Command Prompt”.

When you start **X1t** from the command line (as opposed to clicking on the icon), you can specify a number of arguments. For example, the command

```
x1t -i my_script -d my_thermo.tdat
```

causes **X1t** to read input commands from a file “my_script”, and to use “my_thermo.tdat” as the thermodynamic database.

The following options are available from the command line:

<code>-cd</code>	Change the working directory to the directory containing the input script specified with the <code>-i</code> option.
<code>-nocd</code>	Do not change the working directory.
<code><input_script></code> <code>-i <input_script></code>	Set a file from which to read input commands.
<code>-gtd <gtdata_dir></code>	Set directory to search for thermodynamic datasets.
<code>-cond <cond_data></code>	Set the dataset for calculating electrical conductivity.
<code>-d <thermo_data></code>	Set the thermodynamic dataset.
<code>-iso <isotope_data></code>	Set the file of isotope fractionation factors to be used.
<code>-s <surface_data></code>	Set a dataset of surface sorption reactions.

Using X2t

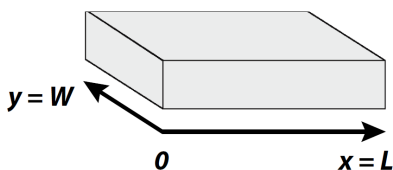
X2t is a program for tracing reactive transport models in two dimensions. It is similar in many ways to the one-dimensional model **X1t**. You should read the previous two chapters of this guide, [Modeling Overview](#) and [Using X1t](#), before beginning this chapter.

The *GWB Reference Manual* contains additional details of the various commands you can use to configure **X1t** and **X2t**.

4.1 Defining the domain

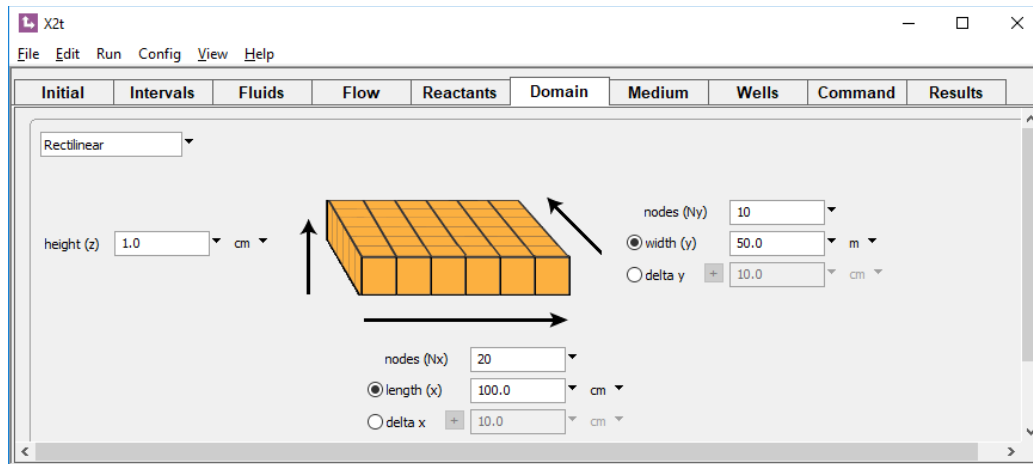
You can configure **X2t** in terms of either a rectilinear or axisymmetric domain. The program divides either type of domain into N_x nodal blocks along x , and N_y blocks along y .

A rectilinear **X2t** domain extends from $x=0$ at its left side to $x=L$ at its right, and $y=0$ at its bottom to $y=W$ at its top.



You set the length L and width W and the numbers of nodal blocks N_x and N_y on the **Domain** pane,

112 *GWB Reactive Transport Modeling*



or with the `length`, `width`, `Nx`, and `Ny` commands. The settings above define a 100 m \times 50 m domain composed of 200 nodal blocks, each of which is 5 m \times 5 m (i.e., Δx and Δy are both 5 m).

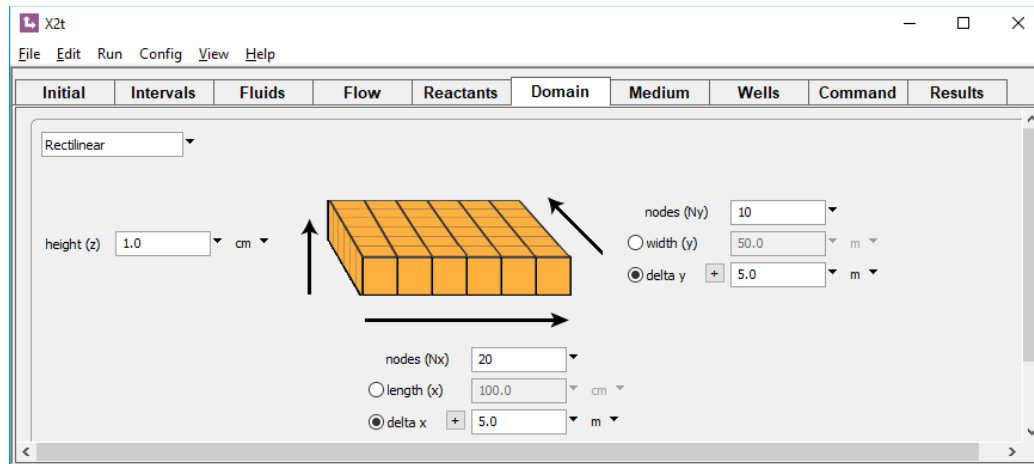
The commands

```
length = 100 m
width = 50 m
Nx = 20
Ny = 10
```

configure the same domain.

You can set the height of the domain (the dimension along z) on the same pane, or with the `height` command. Simulations are commonly configured, however, so that this setting has little effect on the calculation results.

You can alternatively specify Δx , Δy , N_x , and N_y , and have the program calculate the domain length and width, L and W . The settings pictured below



or the commands

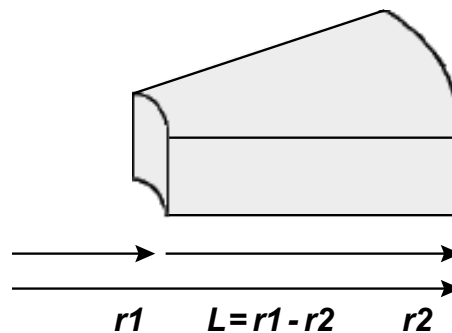
```

deltax = 5 m
deltay = 5 m
Nx = 20
Ny = 10

```

are equivalent to the previous example.

An axisymmetrical domain extends from its small-radius end, at which $r = r1$, to a large-radius end, where $r = r2$.



The length L of an axisymmetrical domain, then, is $r1 - r2$. Normally, the domain is oriented with the small-radius end to the left, but you can reverse its sense.

You define the coordinates $r1$ and $r2$ of an axisymmetrical domain, as well as $\Delta\theta$, the angle of divergence. If you are working from the command line, use the `radial` (or `wedge`) command with keywords `r1`, `r2`, and `angle`. You set values for $r1$ and $r2$ in units of length, and $\Delta\theta$ in radians or degrees. As before, the `Nx` and `Ny` commands define the number of nodal blocks.

You can alternatively set the node spacing Δr with the `deltar` command and let the program determine the domain length L from the values you set for N_x and $r1$, much as you would for a rectilinear domain.

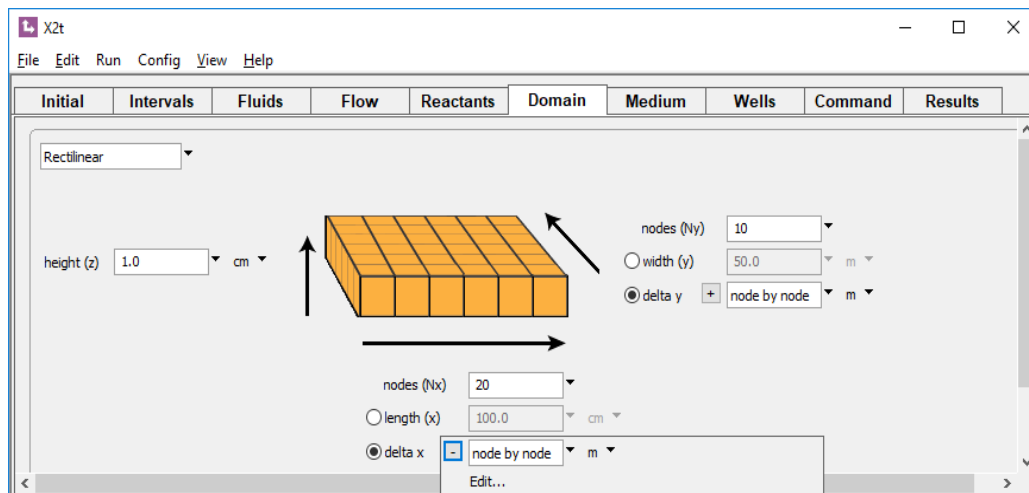
Once you have set an axisymmetrical domain, you can return to rectilinear coordinates on the pane, or with the command

```
radial off
```

To reverse the sense of an axisymmetrical domain, select “Reverse radial” on the pane, or enter the command

```
radial reverse
```

You can prescribe variable nodal block spacings by taking advantage of the fact that **X2t** carries Δx and Δy as field variables (see the [Heterogeneity](#) appendix to this guide). On the **Domain** pane, select the “delta x” button, then click on the **+** next to it. Choose “node by node” from the pulldown and click “Edit...”



to bring up the node by node editor. Entering in values for Δx ,

Field variable: node by node

Populate Edit

	1	2	3	4	5	6	7	8	9	
0	10	10	5	2	2	5	10	10	20	^
										v

< >

OK Apply Cancel Reset

and, doing the same for Δy ,

Field variable: node by node

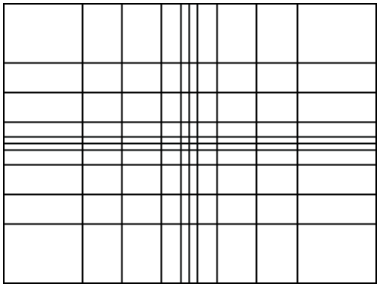
Populate Edit

	0	
0	20	
1	10	
2	10	
3	5	
4	2	
5	2	
6	5	
7	10	
8	10	
9	20	

< >

OK Apply Cancel Reset

defines the grid



The commands

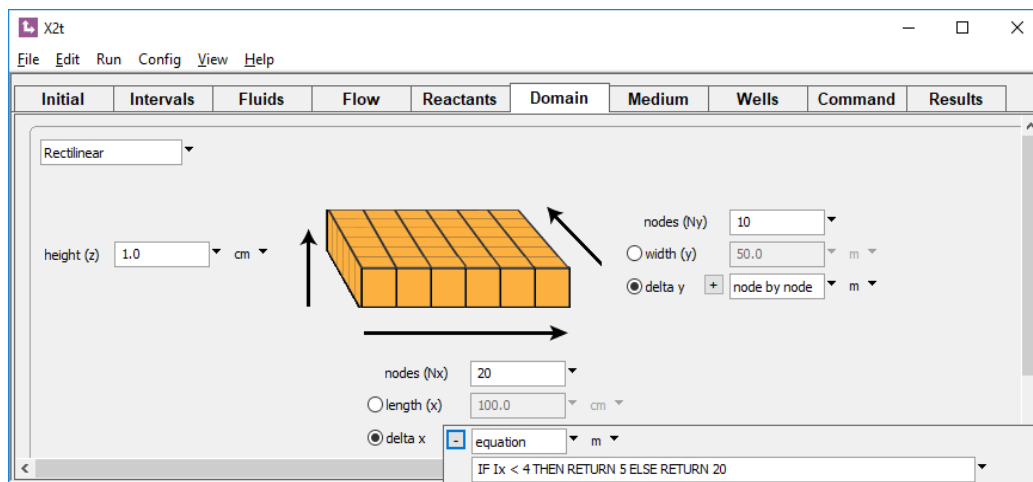
```

deltax = {20 , 10 , 10 , 5 , 2 , 2 , 5 , 10 , 10 , 20} m
deltay = {20 | 10 | 10 | 5 | 2 | 2 | 5 | 10 | 10 | 20} m
Nx = 10
Ny = 10

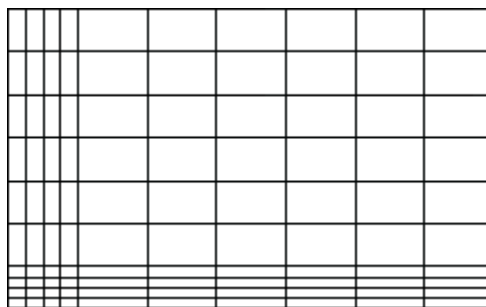
```

configure the same grid.

As another example, you can select “equation” from the pulldowns for “delta x” and “delta y” and type in a simple expression



to produce a grid that looks like





The commands

```
deltax = eqn "IF Ix < 4 THEN RETURN 5 ELSE RETURN 20" m
deltay = eqn "IF Jy < 4 THEN RETURN 5 ELSE RETURN 20" m
Nx = 10
Ny = 10
```

produce the same grid.

4.2 Wells

You can set any number of wells in an **X2t** simulation, and each well can operate over the entire simulation, or only parts of it. The program accounts for the flow into and out of wells when it determines the flow regime within the domain.

You create and define wells on the **Wells** pane. Click on the  button to define a new well; to remove a well, click on the well's label (e.g., "my_well") and then press the  key. Or, on the **Command** pane, use the `well` command

```
well my_well
```

to create a well, and the `remove` command

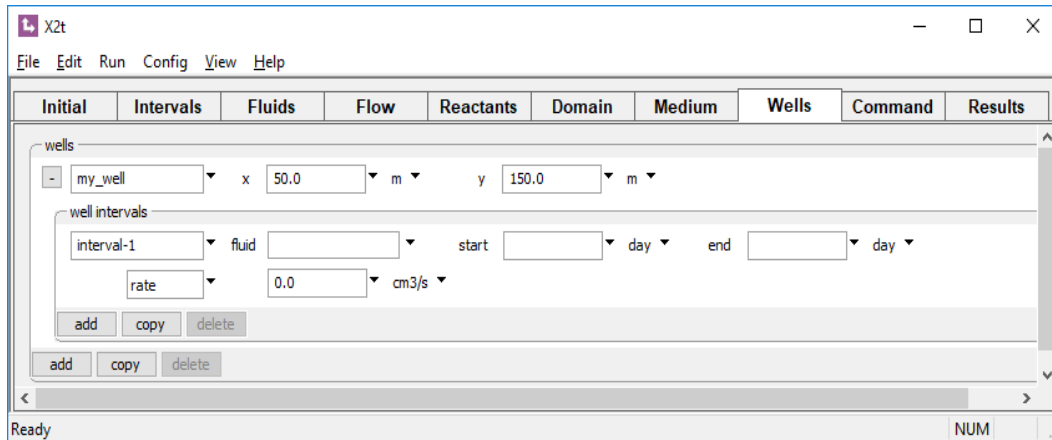
```
remove well my_well
```

to delete a specific well, or

```
remove wells
```

to delete all the wells that have been created.

You set the well's position within the domain in x-y coordinates. You can use the GUI

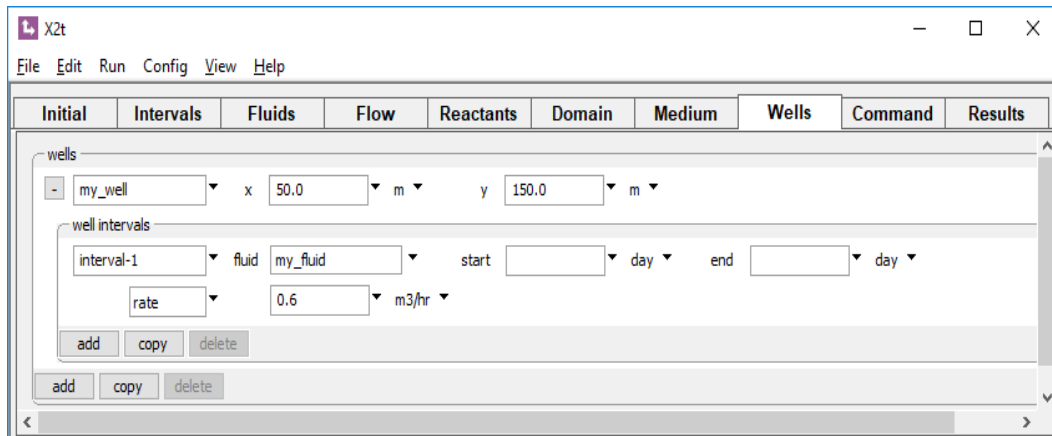


or the command

```
well my_well x = 50 m, y = 150 m
```

The coordinates must lie between 0 and the length and width of the domain. **X2t** determines the nodal block within which the well falls; the actual position carried in the calculation will be that of the corresponding nodal point.

For each injecting well, you set a fluid to inject and an injection rate on the **Wells** pane



or with the equivalent command

```
well my_well fluid = my_fluid, rate = 0.6 m3/hr
```


You can set the injection rate in a variety of units: cm^3 , m^3 , l, gal, or ft^3 per time units (s, min, hr, day, yr, or m.y.). If you omit the well name,

```
well fluid = my_fluid, rate = 0.6 m3/hr
```

the program defaults to the most recently referenced well, or creates a well, if none exists.

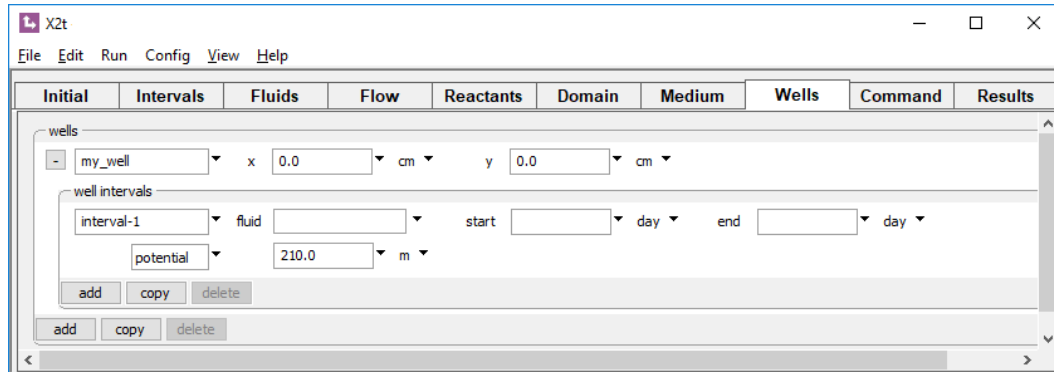
A negative injection rate defines a producing well. For example, the command

```
well rate = -200 m3/day
```

sets the rate of groundwater production from the most recently referenced well to 200 m^3/day . You need not assign a fluid composition in the case of a producing well, since the fluid is taken from the domain, rather than injected into it.

When you assign a well's flow rate, **X2t** incorporates that rate into the calculation of the flow field within the domain. The flow calculation provides a value of hydraulic potential corresponding to the well's position within the flow regime. You can alternatively set the hydraulic head or potential at the well and have **X2t** calculate the well's injection or production rate.

For example,



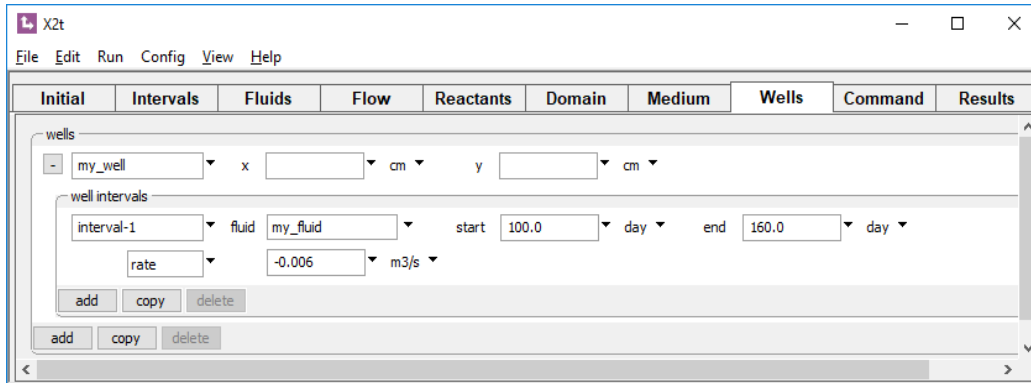
defines the water level in “my_well”. If the head in the neighboring nodes is higher than this value, the well will produce water; if lower, water will flow into the domain. You can set the head in units of mm, cm, m, km, in, ft, or mi, and hydraulic potential as Pa, Mpa, atm, bar, or psi. The command

```
well my_well head 210 m
```

is equivalent.

120 *GWB Reactive Transport Modeling*

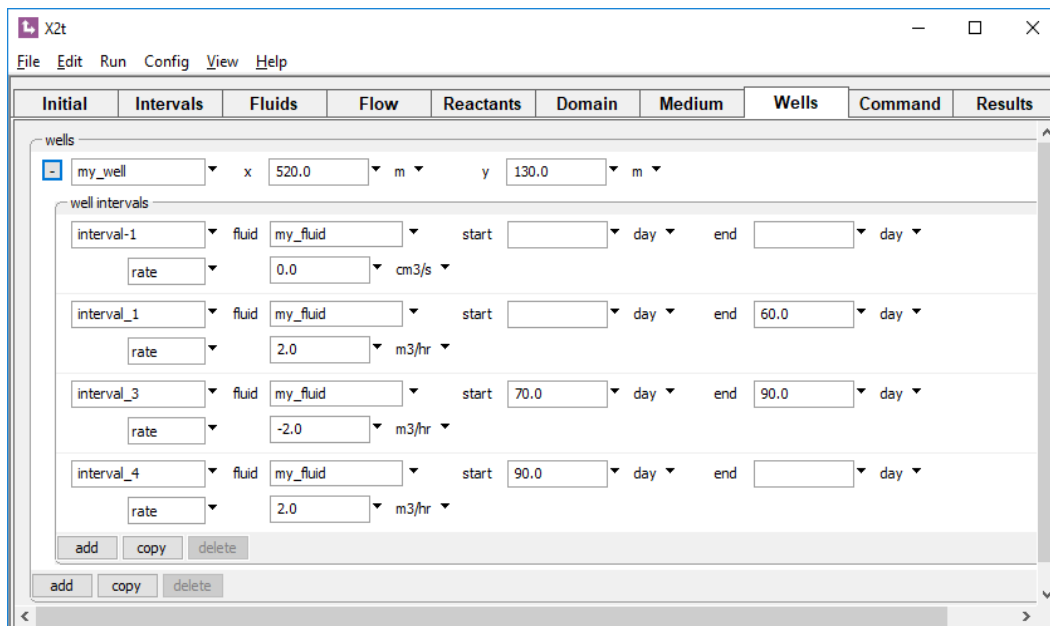
You can set a well to operate within only a specific interval during a simulation. For example,



or the command

```
well my well start = 100 days, end = 160 days, rate = -.006 m3/s
```

sets a well that produces water only between 100 days and 160 days. You can, in fact, set any number of well intervals:

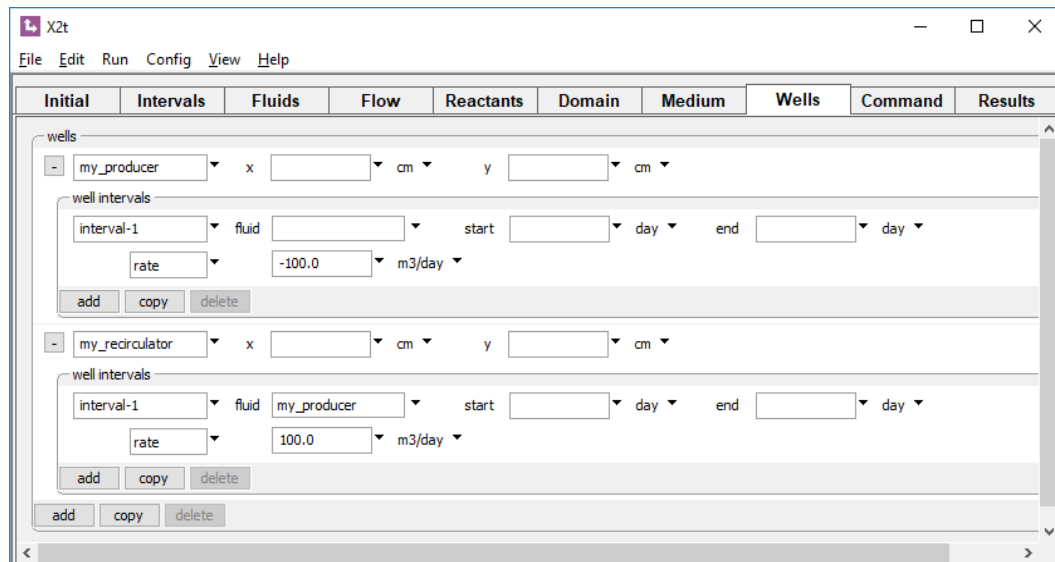


or

```
well my_well x = 520 m, y = 130 m, fluid = my_fluid
well interval my_interval-1 end = 60 days, rate = 2 m3/hr
well interval my_interval-2 start = 70 days, end = 90 days, rate = -2 m3/hr
well interval my_interval-3 start = 90 days, rate = 2 m3/hr
```

In this example, well “my_well” injects “my_fluid” from the start of the simulation to 60 days, remains idle for 10 days, produces formation fluid for the next 20 days, and then resumes injecting “my_fluid” until the end of the simulation.

To configure a recirculation well,



the well “my_recirculator” injects water extracted from the domain by well “my_producer”. The commands

```
well my_producer rate = -100 m3/day
well my_recirculator rate = 100 m3/day, fluid = "my_producer"
```

set up the same configuration.

4.3 Calculating the flow field

You can either have **X2t** calculate the groundwater flow field within the domain at each time step from the boundary conditions and well settings you supply, accounting for the evolving medium permeability and fluid viscosity, or you can import the flow field calculated by another program, as described in the next two sections of this chapter.

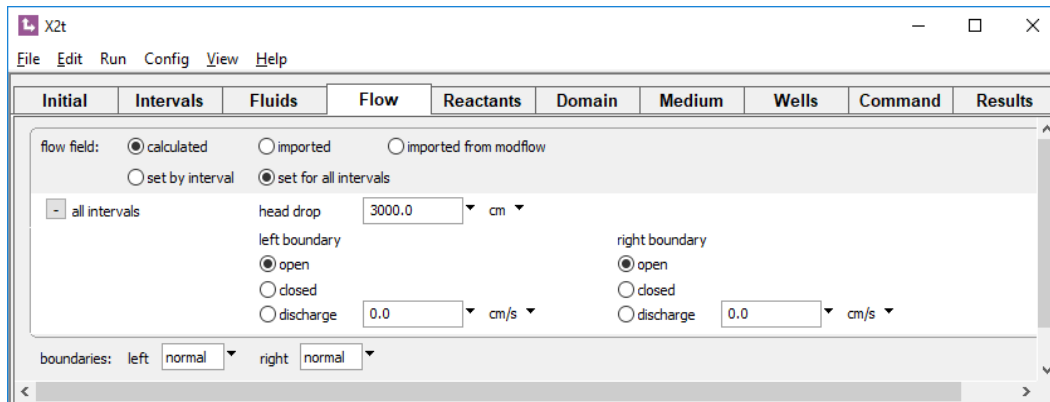
If you have the program figure the flow field, you set the boundary conditions on the **Flow** pane, or on the **Command** pane using various commands: `head_drop`, `pot_drop`, `left`, `right`, and `discharge`. You set the drop in hydraulic head or hydraulic potential across the domain, from left to right. The program converts settings for hydraulic head to hydraulic potential, assuming a fluid density of 1 g/cm³. A positive drop drives flow to the right, a negative value, to the left.

The left-side and right-side boundaries can each be set as:

- open to flow
- closed to flow (no-flow)
- specified discharge across the bound, positive for flow to the right

A boundary open to flow is set in the simulation to a constant-potential. If the left boundary is open, potential along it is set to the value of the potential drop. Head or potential along an open right boundary is set to 0. A no-flow boundary is a special case of a specified discharge boundary, in which the discharge is set to 0. Boundaries are, by default, open to flow.

To set a flow field arising from the head drop across the domain, you could specify

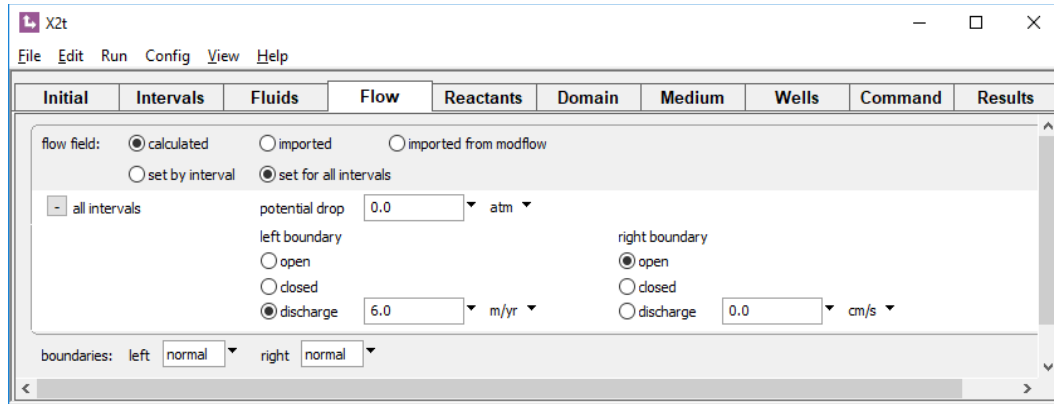


or use a set of commands

```
left = open
right = open
head_drop 3000 cm
```

Note that the first two commands enforce default conditions, and might be omitted, absent previous settings.

To specify a known rate of flow, as a second example, you could specify

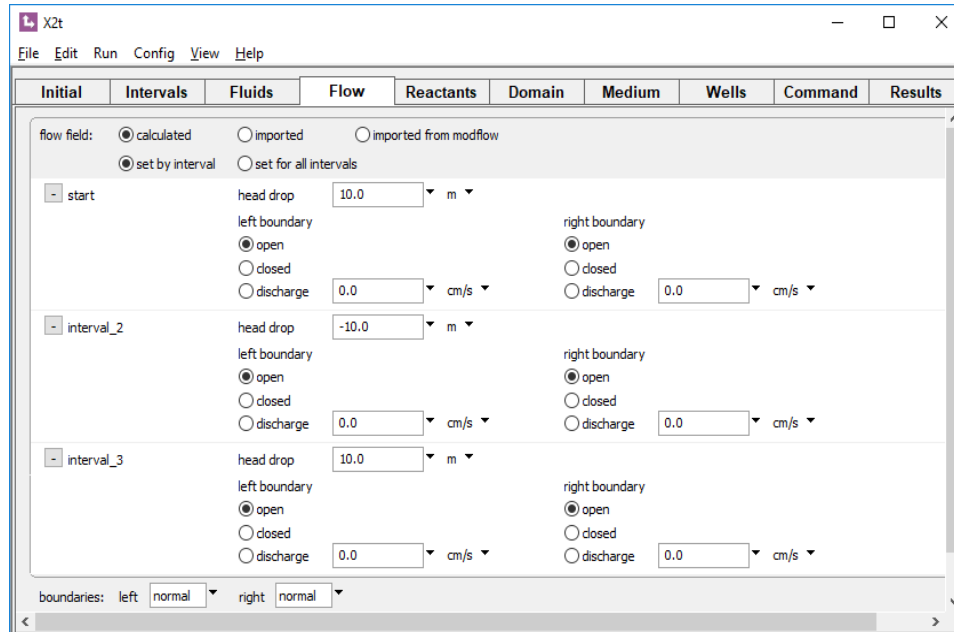


or use a set of commands

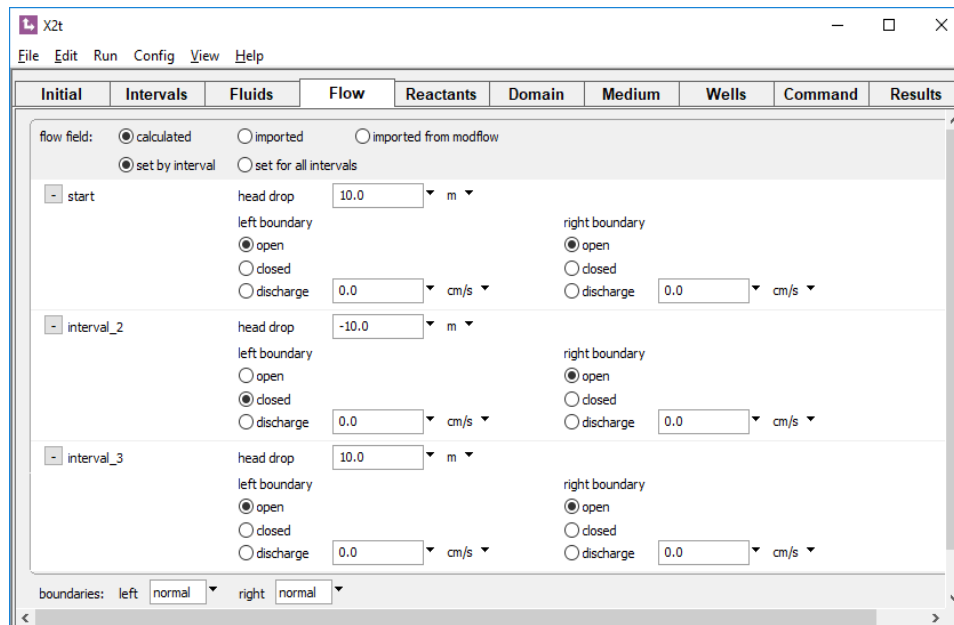
```
right = open
discharge left = 6 m/yr
```

Again, the first command is not necessary, unless the right boundary has previously been set otherwise.

In a simulation of more than one boundary interval, you can specify separate boundary conditions for each interval. In a simulation consisting of three intervals, for example,



sets a reversing flow regime, and



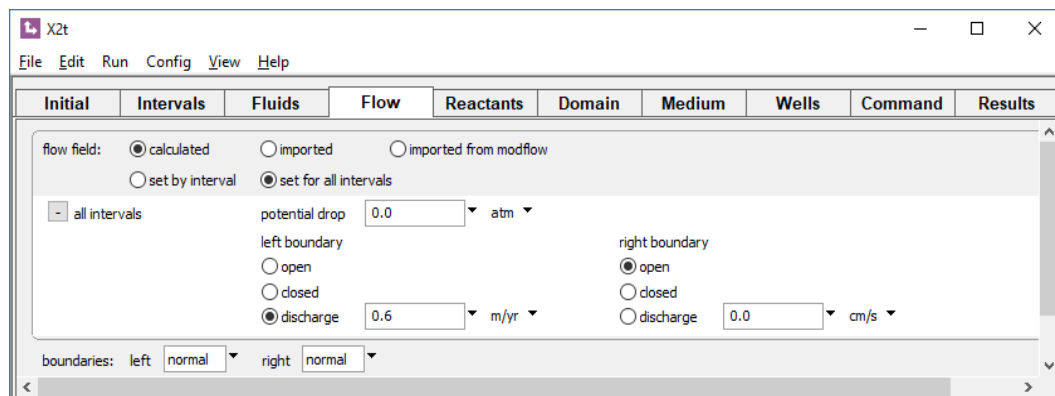
closes and then reopens the left boundary.

The commands

```
head_drop start = 10 m, interval-2 = -10 m, interval-3 = 10 m
and
left start = open, interval-2 = closed, interval-3 = open
```

are equivalent to the configurations shown above, assuming each interval has already been created.

When you use the `discharge`, `head_drop`, and `pot_drop` commands without specifying a simulation interval,



your settings apply to all the intervals in a simulation. In this example, a discharge of 0.6 m yr^{-1} across the left boundary is applied to the left side of all of the boundary intervals in the simulation, regardless of how they might have been set by previous commands.

The bottom and top boundaries to the domain, when **X2t** calculates the flow field, are closed to flow. This restriction does not apply, however, to imported flow fields, as discussed in the subsequent sections. **X2t**, furthermore, does not currently account for the effect of buoyant forces in driving groundwater flow. To model convection, you need to import the flow field.

You can also use the `discharge`, `head_drop`, and `pot_drop` commands to assign a fluid, or fluids, to a boundary interval:

```
discharge start 0.6 m/yr, fluid = fresh_water
discharge interval-2 0.6 m/yr, fluid = seawater
discharge interval-3 0 m/yr left fluid = fresh_water,
    right fluid = seawater
```

This syntax provides an alternative to assigning boundary fluids on the `interval` command.

As in any flow field calculation, it is possible to prescribe boundary conditions that are physically impossible. In an **X2t** simulation that includes no wells, for example, the command

```
discharge left = 10 m/yr, right = -10 m/yr
```

yields an impossible result, because fluid enters the domain along both the left and right sides, but nowhere leaves it. In such cases, **X2t** will report that it cannot solve for the flow field and abandon the simulation. It is best practice to set the discharge across either the left boundary, or the right, but not both.

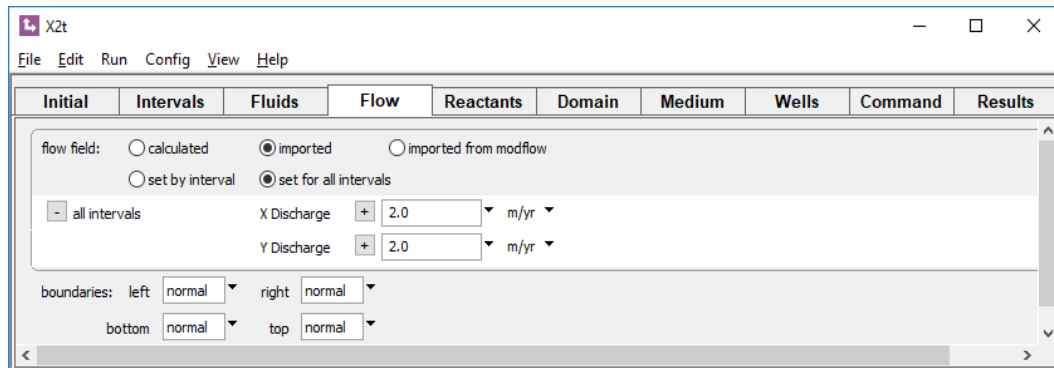
4.4 Importing the flow field

Rather than having **X2t** calculate the groundwater flow field from the boundary conditions and medium properties you supply, you can set it to import the flow field determined by another program. You can also have **X2t** import the results of a simulation made using the popular **MODFLOW** software, using a separate procedure described in the next section, [Importing from MODFLOW](#).

A flow field imported from a program other than **MODFLOW** remains constant in the simulation over the course of a reaction interval. It does not vary in response to changes in medium permeability or fluid viscosity. Unlike a calculated flow field, in which the top and bottom bounds are held closed to flow, groundwater in an imported flow field can enter or leave the domain across any of the four boundaries.

X2t carries an imported flow field internally as field variables, one for discharge along x , and another for y . Field variables are described in the [Heterogeneity](#) appendix to this guide, and you should review this appendix before continuing. You set an imported flow field on the **Domain** pane, by clicking on “Imported”, or on the **Command** pane, using the `discharge` command.

You can define discharge using any of the options associated with field variables, such as with a constant value or an equation. In this example,

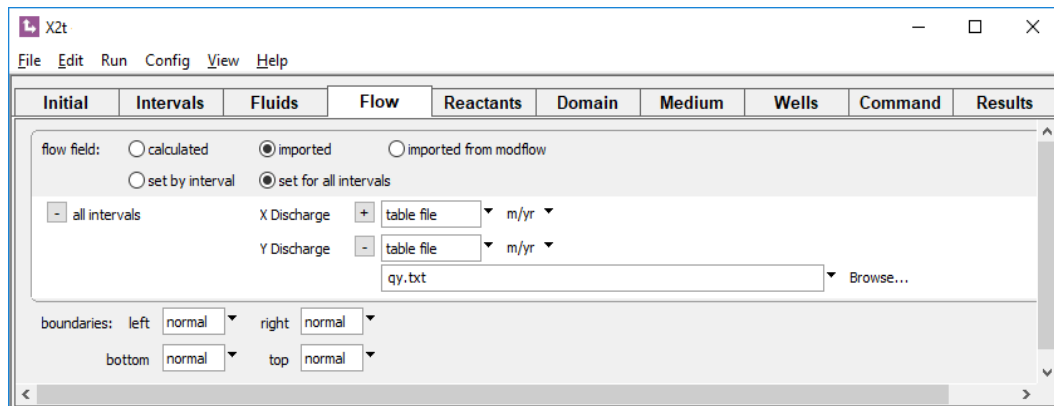


flow moves diagonally from the lower left to the upper right of the domain. The command

```
discharge x = 2 m/yr, y = 2 m/yr
```

is equivalent.

Most commonly, however, the “table” option



provides the most convenient means for defining the flow field.

Use a command like

```
discharge x = qx.txt m/yr, y = qy.txt m/yr
```

to read in the tables.

From the results of running the external program, you can prepare tables in text format for specific discharge along x and y . The tables contain values representing groundwater flow among the nodal blocks in the domain, in any of the units **X2t** recognizes. You can create one table containing values for discharge along x , and a second for flow along y , and you can create separate tables for each reaction interval. **X2t** reads the tables at the

start of the simulation and uses them to set the rates and direction of groundwater flow within the domain.

Keep in mind that the tables contain values for groundwater discharge across nodal block boundaries, from one node to the next. The values are indexed according to the nodal block boundary, rather than the nodal block, in question, as described in the [Node indexing](#) section of the **Modeling Overview** chapter in this guide.

In a table of x discharges, the first value on a line represents flow across the left boundary into (or out of) a row's left-most nodal block. The second value is flow from the first to the second node, the third gives flow from the second to the third, and so on. The final value is flow across the right boundary. If in a domain there are 10 nodes along x , each line in the table should contain 11 values. The first line in the table represents flow along the bottom row of nodes, the second, the row above, and so on. For an $N_x \times N_y$ domain, then, the x -direction table should contain N_y lines, each line containing $N_x + 1$ values.

A table of y discharges, in contrast, should contain $N_y + 1$ lines, N_x values per line. The first line represents flow across the lower boundary into the bottom row of nodal blocks, the second line gives flow from this row to the row above, and so on until the last line, which is flow across the top boundary. On each line, the first value is flow within the left-most column of nodes, and each subsequent value is associated with the adjacent column to the right.

The boundary conditions for the simulation are implicit in the imported flow field and do not need to be set explicitly. If the values on the first column of the x discharge dataset are all 0, for example, the left side of the domain is a no-flow boundary. Where the values are positive, groundwater enters the domain, and where they are negative, it leaves. Similarly, the last (more precisely, the $N_x + 1^{\text{th}}$) column in the x dataset describes the right-side bound, and the first and last ($N_y + 1^{\text{th}}$) lines of the y dataset give the bottom and top boundary conditions.

4.5 Importing from MODFLOW

If you have calculated the groundwater flow pattern using the **MODFLOW** software, you can import the calculation results into **X2t** directly, rather than preparing tables as described in the previous section. For complete information on this option, refer to the [Importing From MODFLOW](#) appendix to this guide.

MODFLOW is a widely used groundwater flow model originally developed by the U.S. Geological Survey; it is available in public domain and commercial versions. In contrast to flow fields imported according to the general procedure described in the previous section, **X2t** will account for variation in the groundwater flow pattern over the course of the run if you have traced a transient simulation in **MODFLOW**.

Begin in **MODFLOW** by running either a steady-state or transient groundwater flow simulation, following the guidelines described in the appendix. When the run is complete, start **X2t** and go to the **Flow** pane, click on “imported from MODFLOW”, then click on “Browse...” and select the discretization file produced by your **MODFLOW** run. Alternatively, you can use the `modflow` command on the **Command** pane to point to the discretization file.

X2t will read in this file and the associated budget file and use the information in them to configure the domain and groundwater flow pattern to be used in your reactive transport simulation.

4.6 Mass transport

X2t, like **X1t**, accounts for mass transport by molecular diffusion, hydrodynamic dispersion, and groundwater advection. The dispersive fluxes (mol/cm²/s), which account for the effects of diffusion as well as dispersion, along the principal coordinates are calculated according to

$$q_{D_x} = -\phi D_{xx} \frac{\partial C}{\partial x} - \phi D_{xy} \frac{\partial C}{\partial y} \quad (4.1)$$

$$q_{D_y} = -\phi D_{yx} \frac{\partial C}{\partial x} - \phi D_{yy} \frac{\partial C}{\partial y} \quad (4.2)$$

where ϕ is sediment porosity, D_{ij} are the entries in dispersion coefficient tensor (cm²/s), and C is concentration (mol/cm³). Terms D_{xx} and D_{yy} in the dispersion tensor are calculated from the relations

$$D_{xx} = D^* + \alpha_L \frac{v_x^2}{|v|} + \alpha_T \frac{v_y^2}{|v|} \quad (4.3)$$

$$D_{yy} = D^* + \alpha_L \frac{v_y^2}{|v|} + \alpha_T \frac{v_x^2}{|v|} \quad (4.4)$$

where D^* is the diffusion coefficient (cm²/s), α_L and α_T are the longitudinal and transverse dispersivity (cm), v_x and v_y are the components of the groundwater velocity (cm/s), and v is the magnitude of the velocity (cm/s).

130 *GWB Reactive Transport Modeling*

You set D^* and the dispersivities as field variables on the **Medium** pane,

The screenshot shows the X2t software interface with the **Medium** pane selected. The pane is divided into several sections:

- medium properties:**
 - diffusion coefficient: $1.0\text{e-}06$ cm²/s (Callout: Diffusion coefficient for the porous medium)
 - porosity: (empty field)
 - inert volume: 0.0 cm³
 - thermal conductivity: 0.004 cal/cm/s/C
 - heat capacity: cpw (fluid) 1.0 cal/g/C, cpr (minerals) 0.2 cal/g/C
 - internal heat source: temperature: minimum (empty) °C, maximum (empty) °C, 0.0 cal/cm³/s
- mass transport:**
 - ☐ include off-diagonal terms in dispersion tensor
 - longitudinal dispersivity: (empty) cm (Callout: If not set, $\alpha_L = L/100$, $\alpha_T = \alpha_L / 10$)
 - transverse dispersivity: (empty) cm
- permeability:** log kx (darcy) = (A x porosity (volume fraction)) + B
 - A (porosity): 15.0
 - B (intercept): -5.0 darcy
 - kx/ky (anisotropy): 1.0

Buttons for 'add' and 'delete' are located at the bottom of the permeability section.

or with the `diffusion_coeff` and `dispersivity` commands. By default, the program assumes that D^* is 10^{-6} cm²/s, α_L is the length of the domain L divided by 100, and α_T is one-tenth of α_L .

The off-diagonal terms D_{xy} and D_{yx} in the tensor are taken as 0, unless the “off-diag” option is set by checking “include off-diagonal terms in dispersion tensor” on the **Medium** pane,

Check to include D_{xy} and D_{yx}

mass transport	<input checked="" type="checkbox"/>	include off-diagonal terms in dispersion tensor
longitudinal dispersivity	+	<input type="text"/> cm
transverse dispersivity	+	<input type="text"/> cm

or using the `off-diag` command. Either way, they are calculated as

$$D_{xy} = D_{yx} = (\alpha_L - \alpha_T) \frac{v_x v_y}{|v|} \quad (4.5)$$

Note that the off-diagonal terms disappear where flow is aligned along either the x or y axis.

The advective flux q_A of a chemical component along x and y is given as

$$q_{A_x} = q_x C \quad (4.6)$$

$$q_{A_y} = q_y C \quad (4.7)$$

where q_x and q_y are the components of specific discharge and C is concentration.

Like **X1t**, **X2t** figures the limiting time step according to the combined von Neumann and Courant stability conditions, according to

$$\left(\frac{|v_x|}{\Delta x} + \frac{|v_y|}{\Delta y} + \frac{2D_{xx}}{\Delta x^2} + \frac{2D_{yy}}{\Delta y^2} + \frac{4D_{xy}}{\Delta x \Delta y} \right) \Delta t \leq 1 \quad (4.8)$$

The right-most term within parentheses comes into play only when the “off-diag” option is set.

4.7 Heat transfer

In tracing a polythermal run, **X2t** accounts for the transfer of heat within the domain by the combined processes of groundwater advection and heat conduction. Advective heat transfer is the movement of heat along with the flowing groundwater. The program calculates the rate of heat conduction from thermal conductivity and the temperature gradient, according to Fourier's law

$$q_{H_x} = -K_H \frac{\partial T}{\partial x} \quad (4.9)$$

$$q_{H_y} = -K_H \frac{\partial T}{\partial y} \quad (4.10)$$

where q_H is the conductive flux (cal/cm²/s) along the principal directions, and K_H is thermal conductivity (cal/cm/s °C). The rate of heat advection in each direction is a product of specific discharge in the appropriate direction and fluid enthalpy.

In polythermal runs, in addition to the stability constraints for mass transport already described, the program figures a limiting value for the time step as a result of solving the heat transfer equation from the relation

$$\left(\frac{|v_x|}{\Delta x} + \frac{|v_y|}{\Delta y} + \frac{2K_T}{\Delta x^2} + \frac{2K_T}{\Delta y^2} \right) \Delta t \leq 1 \quad (4.11)$$

The program then uses the lesser of the limiting step for mass transport and that for heat transfer to constrain its time marching.

4.8 Example input files

Subsequent sections of this chapter contain examples of applying **X2t**. An input file corresponding to each example is available in the GWB installation directory (e.g., “\Program Files\GWB”) under the subdirectory “Script”.

The example files are

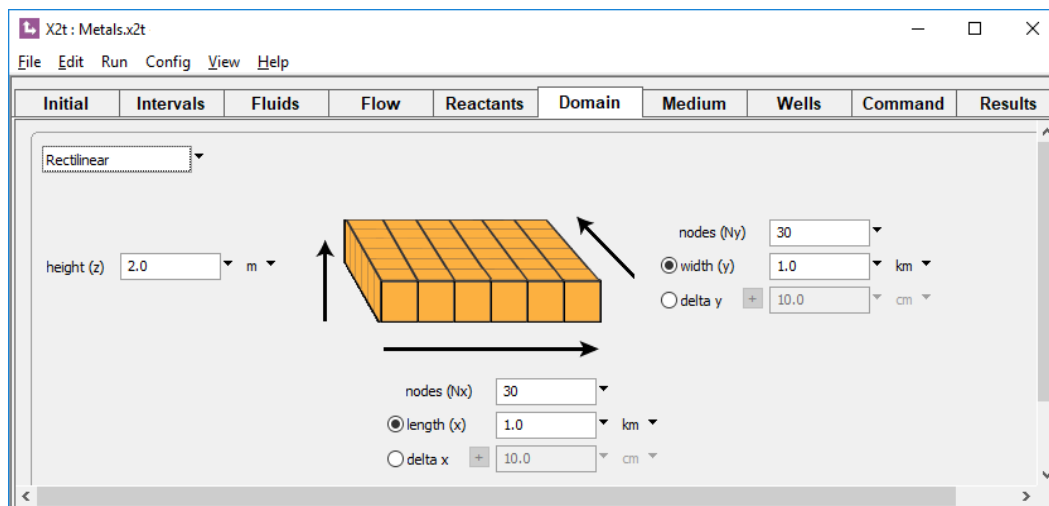
- “Metals.x2t” Metals contamination of an aquifer
- “Steam_2D.x2t” Steam flood

4.9 Example: metals contamination of an aquifer

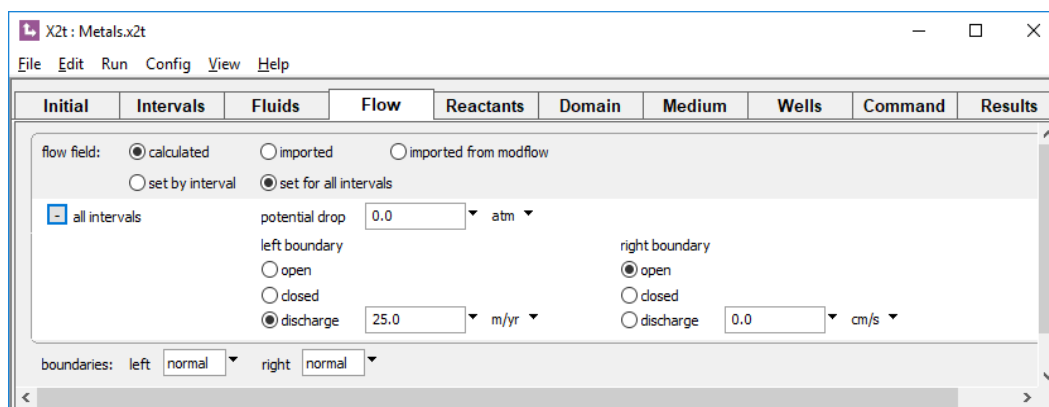
Expanding on the one-dimensional examples in the previous chapter (see [Example: Pb contamination](#) and [Example: groundwater chromatography](#) in the [Using X1t](#) chapter), we construct a two-dimensional model of the contamination of an aquifer by heavy metals and its subsequent remediation. As before, we consider the metals Pb^{++} , Cu^{++} , and Ni^{++} , and use the modified two-layer model of surface complexation, as presented by Dzombak and Morel (1990), to account for ion sorption.

After 10 years, the sources are controlled and a remediation well is installed down-gradient of the tanks to draw out contaminated groundwater. The remediation well is pumped in the simulation for 10 years.

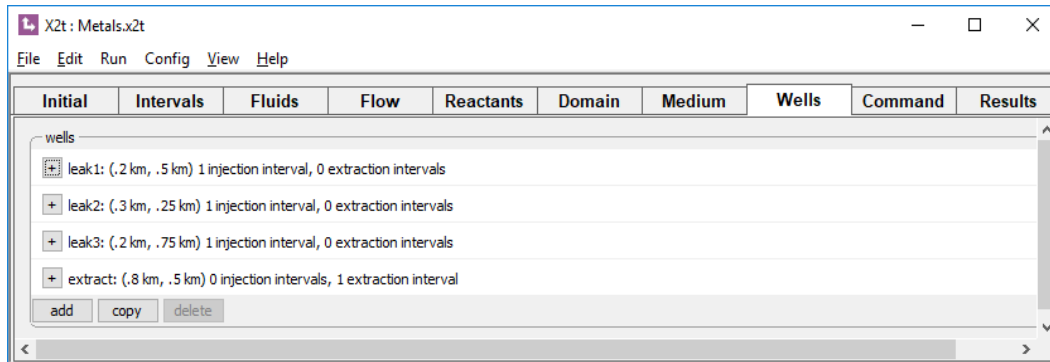
Start X2t by double-clicking on file “Metals.x2t”. On the **Domain** pane, we see the domain is 1 square km, divided into 900 nodal blocks.



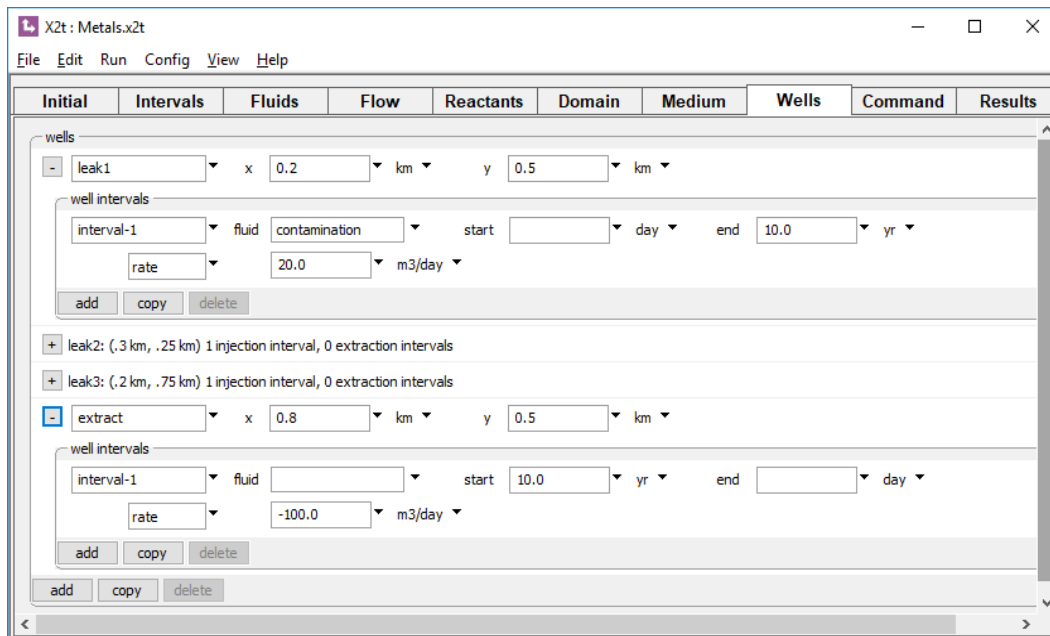
On the **Flow** pane, we see the background discharge is 25 m/yr, left to right



Four wells are defined on the **Wells** pane.



The first three slowly leak the contamination fluid into the aquifer over the first 10 years. The remaining well, at the bottom of the list, is the remediation well. It withdraws water from the aquifer, so the injection rate is set to a negative value.



Fluid is withdrawn from the aquifer in the extract well, but none is injected, so no injectate chemistry needs be specified.

The **Initial** pane specifies a clean fluid.

X2t : Metals.x2t

File Edit Run Config View Help

Initial Intervals Fluids Flow Reactants Domain Medium Wells Command Results

constraints on initial system

H2O		+	1.0	free kg	solvent
Fe(OH)3(ppd)	Fe+++	+	0.01	volume%	
H+		+	6.0	pH	
Na+		+	0.001	molal	
Cl-		+	0.001	molal	charge balance
Pb++		+	0.1	ug/kg	
Cu++		+	0.1	ug/kg	
Ni++		+	0.1	ug/kg	
Br-		+	0.1	ug/kg	
temperature		+	25.0	C	
		+	advanced		

add copy delete

The **Fluids** pane specifies a clean fluid, an ambient, and the contaminated water, laden with the inorganic metal pollutants, as well as a bromide tracer.

X2t : Metals.x2t

File Edit Run Config View Help

Initial Intervals Fluids Flow Reactants Domain Medium Wells Command Results

inlet fluids

+ ambient

- contamination

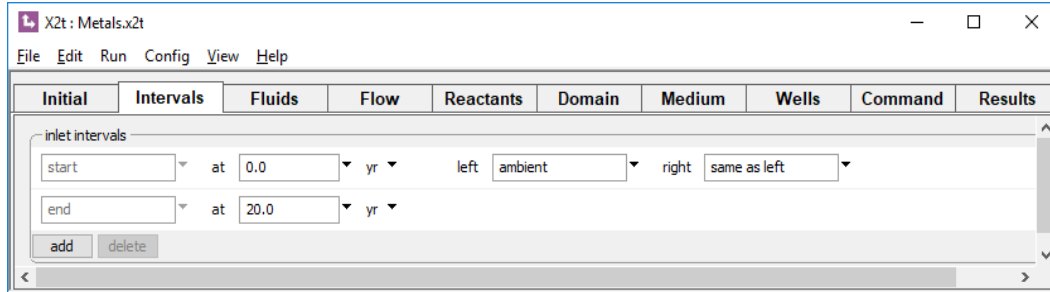
constraints on fluid

H2O		1.0	free kg	solvent
Fe(OH)3(ppd)	Fe+++		free mmol/kg	
H+		6.0	pH	
Na+		0.001	molal	
Cl-		0.001	molal	charge balance
Pb++		0.1	mmolal	
Cu++		0.1	mmolal	
Ni++		0.1	mmolal	
Br-		0.1	mmolal	
temperature		25.0	C	
		+	advanced	

add copy delete

add copy delete

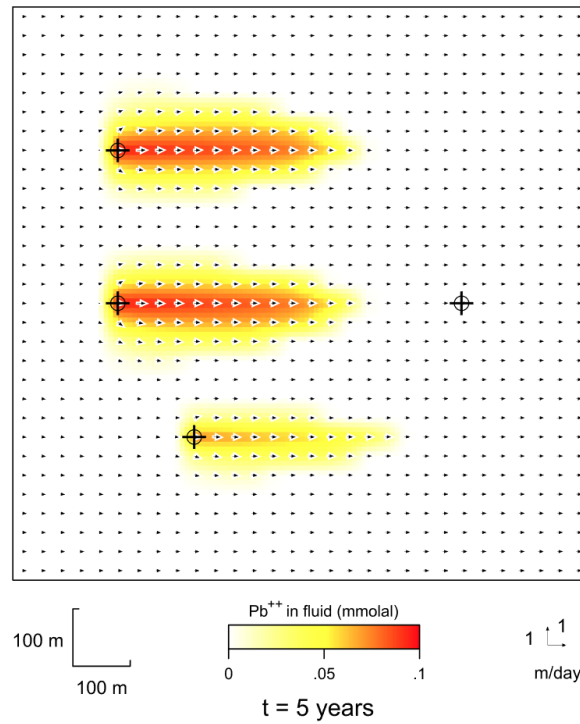
The **Intervals** pane specifies the duration of the simulation and the fluid crossing the left boundary.



You can trace the simulation by selecting **Run → Go**. When the run is complete, clicking on **Plot Results** launches **Xtplot**.

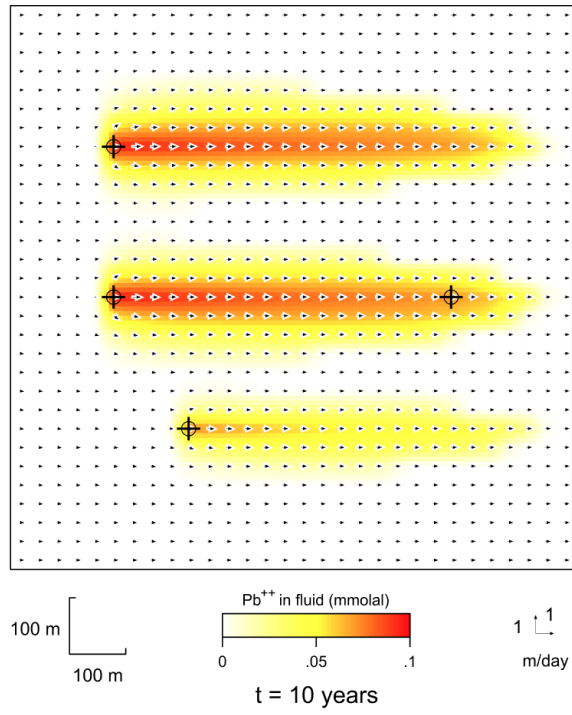
We can use **Xtplot**'s color mapping feature to show how metal concentration in the groundwater evolves across the domain. Select **Map View** under the **Plot** pulldown. Click on **Format → Color Map...** to expose the color mapping dialog. Choose "Components in fluid", "Pb++", and "mmolal" for the variable entries, then click on **Apply**. Under **Format → Quick Toggle...**, select "Velocity Arrows" and "Velocity Scale", and deselect "Contour Lines". Under **Format → Time Step...**, select "5 years".

You should see a plot showing the distribution of Pb⁺⁺ concentration in the groundwater 5 years into the simulation. You can refine the plot by rendering it square (choose the **Plot Area** tab from the **Format → Appearance...** dialog), choosing appropriate units, lengths, and positions for the distance scale (**Format → Distance Scale...**), arrow scale (**Format → Velocity Arrow...**), and various other elements. Your diagram should look something like

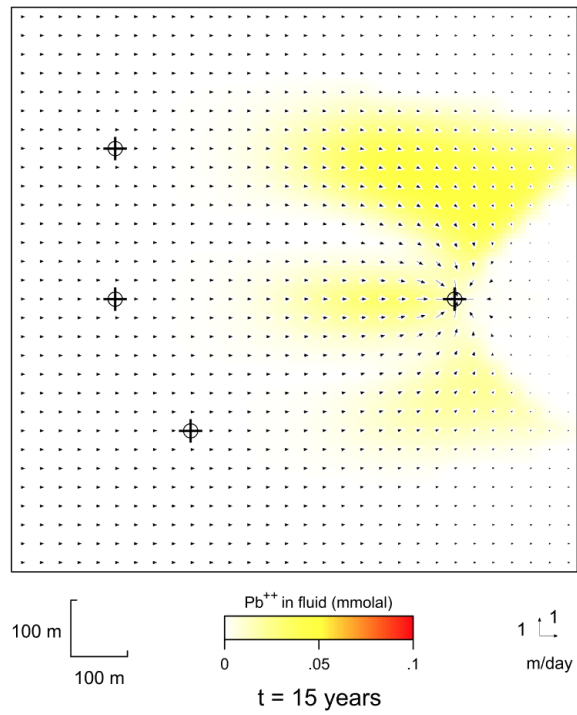


You can represent other variables on this plot using the contouring (**Format → Contour...**) and color masking (**Format → Color Mask...**) features.

You can make similar plots for Cu^{++} and Ni^{++} , as well as Br^- , the conservative tracer. By selecting other time levels under **Format → Time Step...**, you can see how metal concentration evolves over the course of the simulation. At 10 years and 15 years, for example, the results for Pb^{++} look like

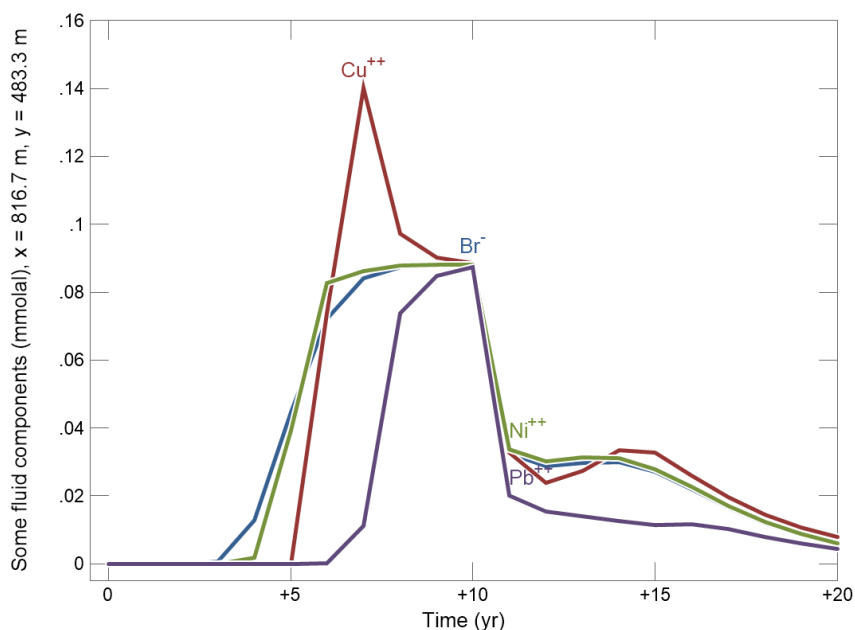


and



To view the entire simulation, select **Format → Animate...**, and then click **Run**. To stop the animation, press the **Esc** key in the main window or **Stop** in the dialog.

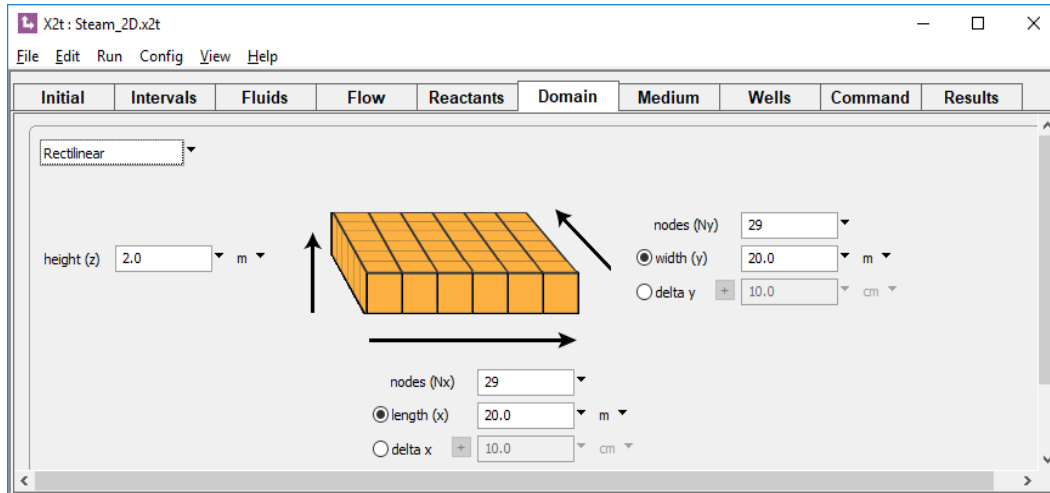
To plot breakthrough curves for the remediation well, select **Plot → XY Plot...**, and on the **X Axis** tab, next to “Display”, select “Time”; on the **Y Axis** tab, next to “Variable type”, select “Components in fluid”. On the **Position** tab, select the node labeled “extract”, which is the remediation well (checking “wells first” conveniently moves nodes containing wells to the top of the list). Then, to show the breakthrough curves for Pb^{++} , Cu^{++} , Ni^{++} , and Br^- , select these four components on the **Y Axis** tab. Using the options on the **X Axis** and **Y Axis** tabs, set units of “years” and “mmolal”. Under the tabbed **Format → Appearance...**, dialog, move to the **Lines and Markers** tab and select “multicolor lines” and “Use line color” for line labels. Your plot should look like this



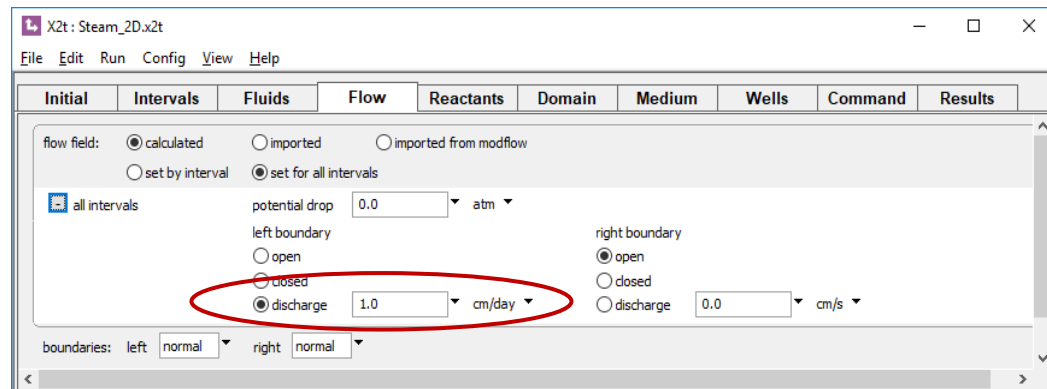
4.10 Example: steam flood

As a further example, we develop a two-dimensional, polythermal model of transport and reaction in a clastic petroleum reservoir during steam flooding, following the one-dimensional example considered previously. In this case, we model the injection of “steam” into a reservoir where the formation fluid is migrating from left to right at a rate of about 1 cm/day. Start by double-clicking on the “Steam_2D.x2t” input file.

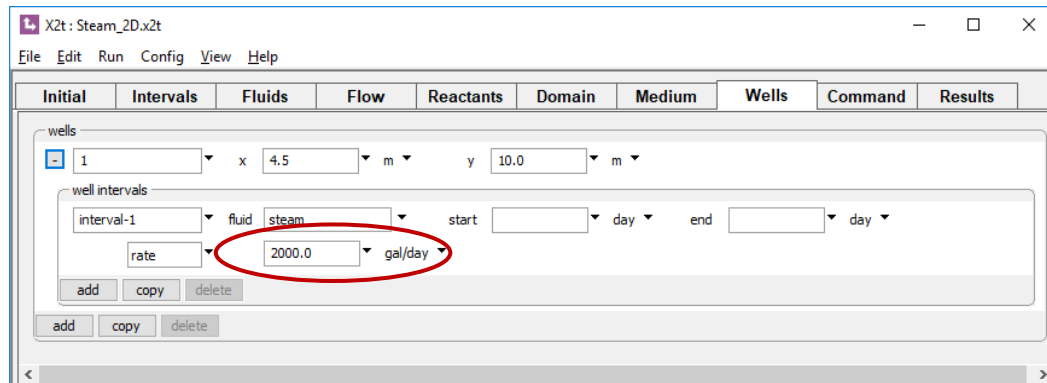
We begin on the **Domain** pane by setting the size and gridding of the domain. The simulation comprises a $20\text{ m} \times 20\text{ m} \times 2\text{ m}$ portion of the formation.



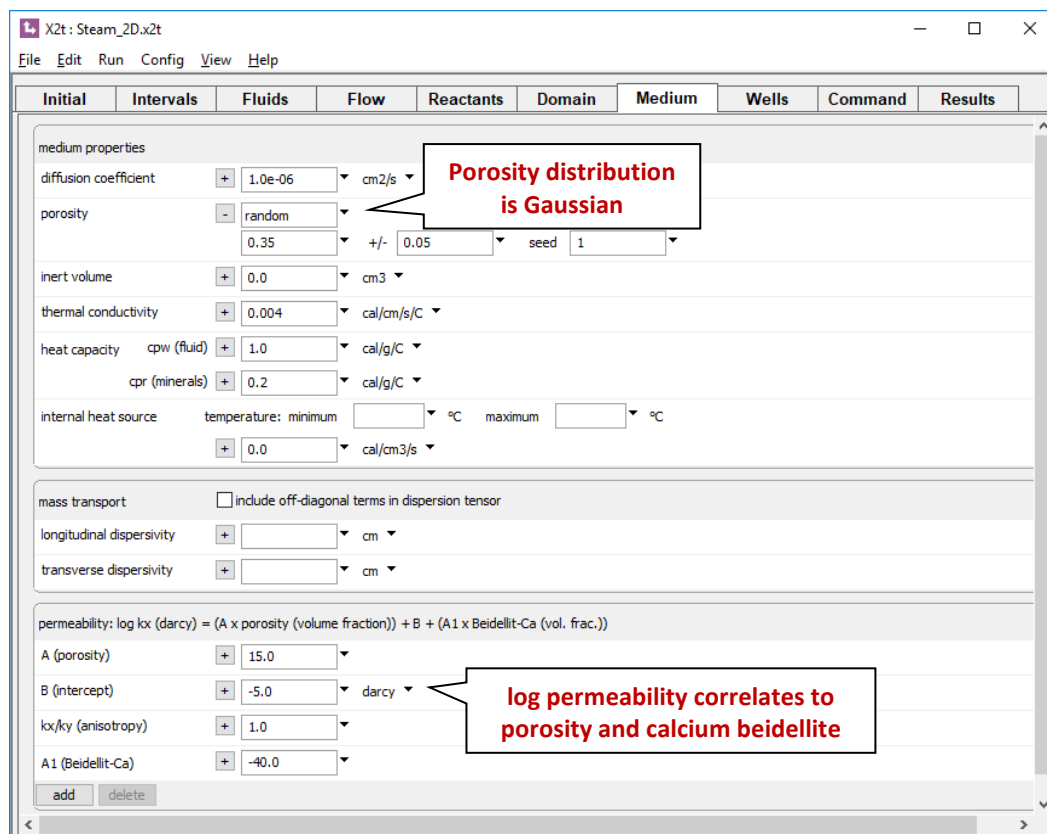
On the **Flow** pane, we define background groundwater flow in the formation by specifying that fluid discharge across the left side of the domain at $1\text{ cm}^3\text{ cm}^{-2}\text{ day}^{-1}$, or 1 cm day^{-1} .



On the **Wells** pane, we further set the position and injection rate for the injecting well. The well is located 4.5 m from the left of the domain and 10 m above the lower bound; it will inject 2000 gallons of “steam” per day.



Moving to the **Medium** pane,



we take advantage of the fact that porosity is carried in the simulation as a field variable by setting it to a random distribution of 35%, with a standard deviation of 5%.

142 *GWB Reactive Transport Modeling*

Following our one-dimensional example, we set permeability to correlate with porosity, as well as the calcium beidellite that will form during the simulation. Since the porosity in the simulation is heterogeneous, permeability will be as well.

On the **Initial** pane,

Component	Chemical Formula	Value	Unit	Description
H2O		1.0	free kg	solvent
Quartz	SiO2(aq)	50.0	volume%	
Calcite	Ca++	5.0	volume%	
Kaolinite	Al+++	5.0	volume%	
H+		6.0	pH	
Na+		1.0	molal	
Cl-		1.0	molal	charge balance
HCO3-		0.01	molal	
temperature		40.0	C	

add copy delete

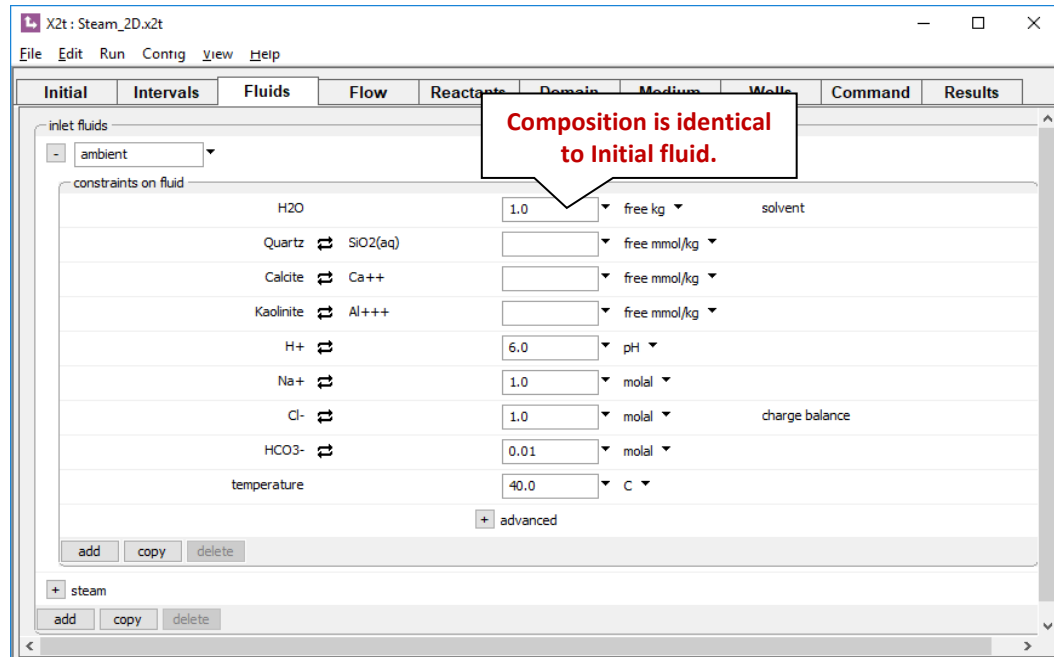
advanced

Initial fluid in equilibrium with quartz, calcite, and kaolinite.

Formation temperature

we set the formation temperature to 40°C and define the formation mineralogy and pore fluid composition as we did in the **X1t** example. The simulation will span 20 days.

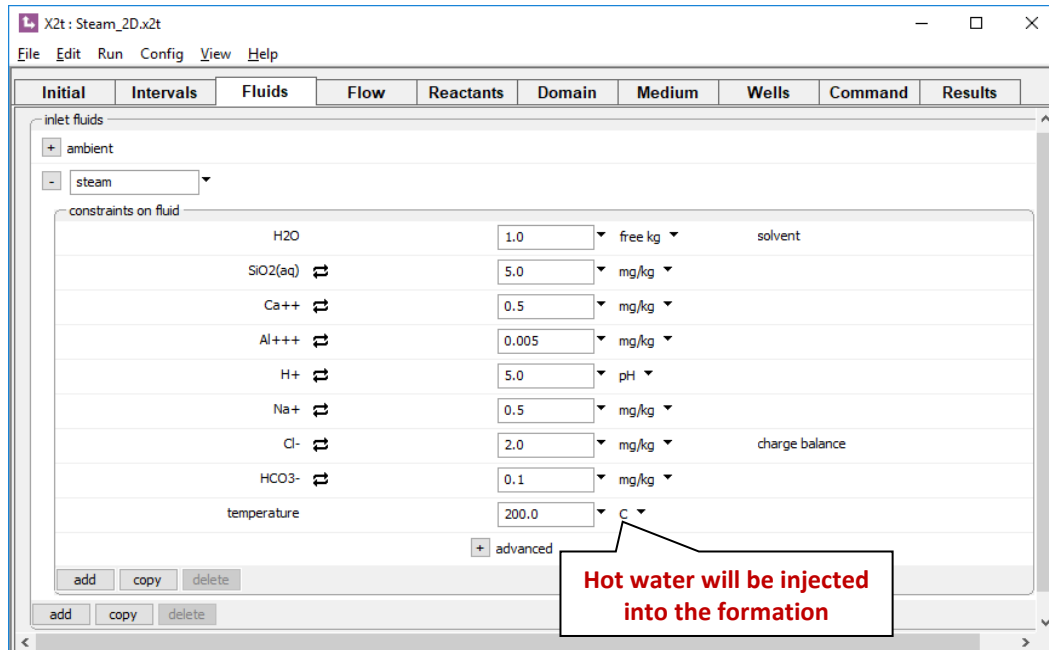
Similarly, we define the fluid that migrates into the reservoir on the **Fluids** pane.



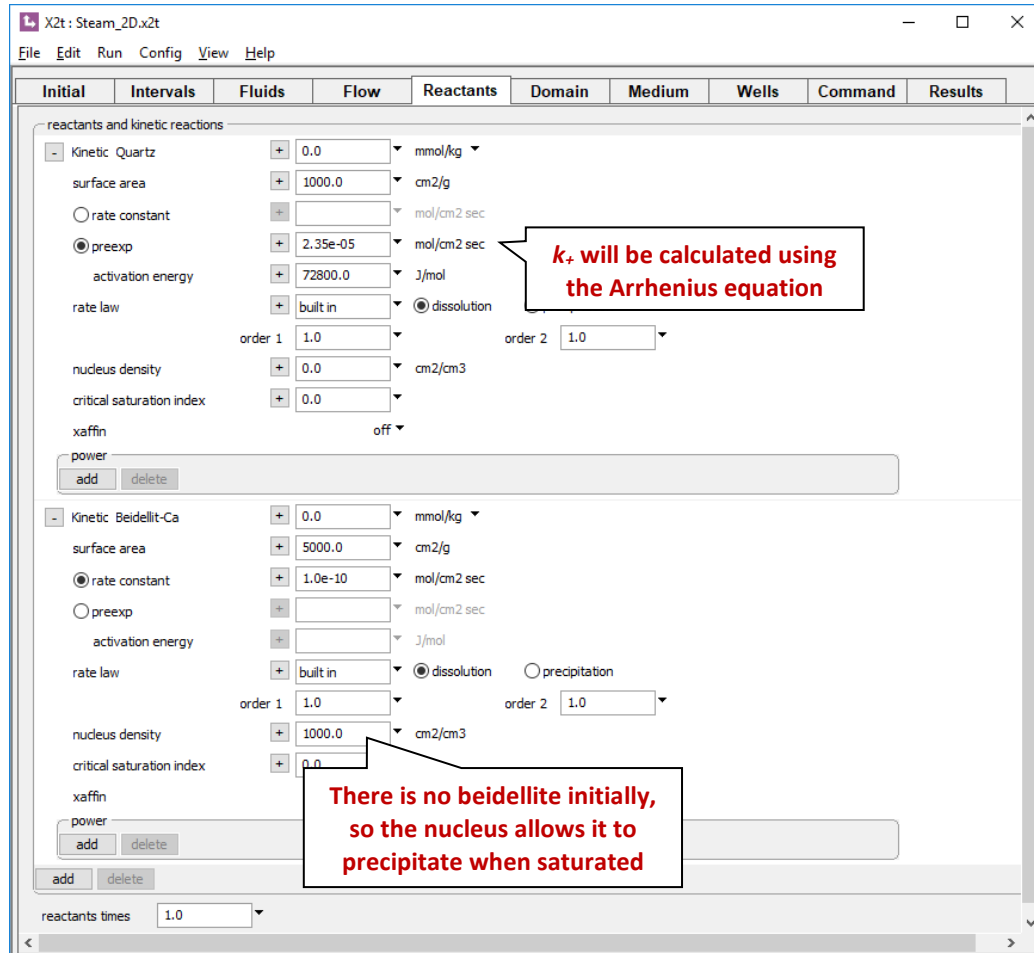
The composition is the same as the **Initial** fluid, but it is not necessary to set the mass of the equilibrium minerals.

144 *GWB Reactive Transport Modeling*

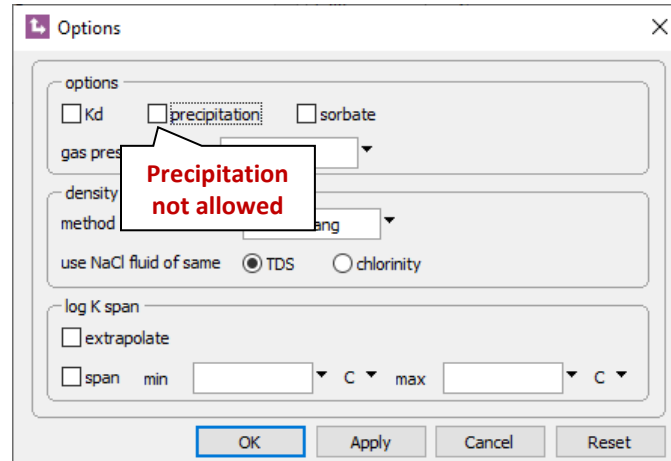
The injectate (“steam”) is fresh water heated to 200°C.



We further define the kinetics of the precipitation reactions on the **Reactants** pane as before.

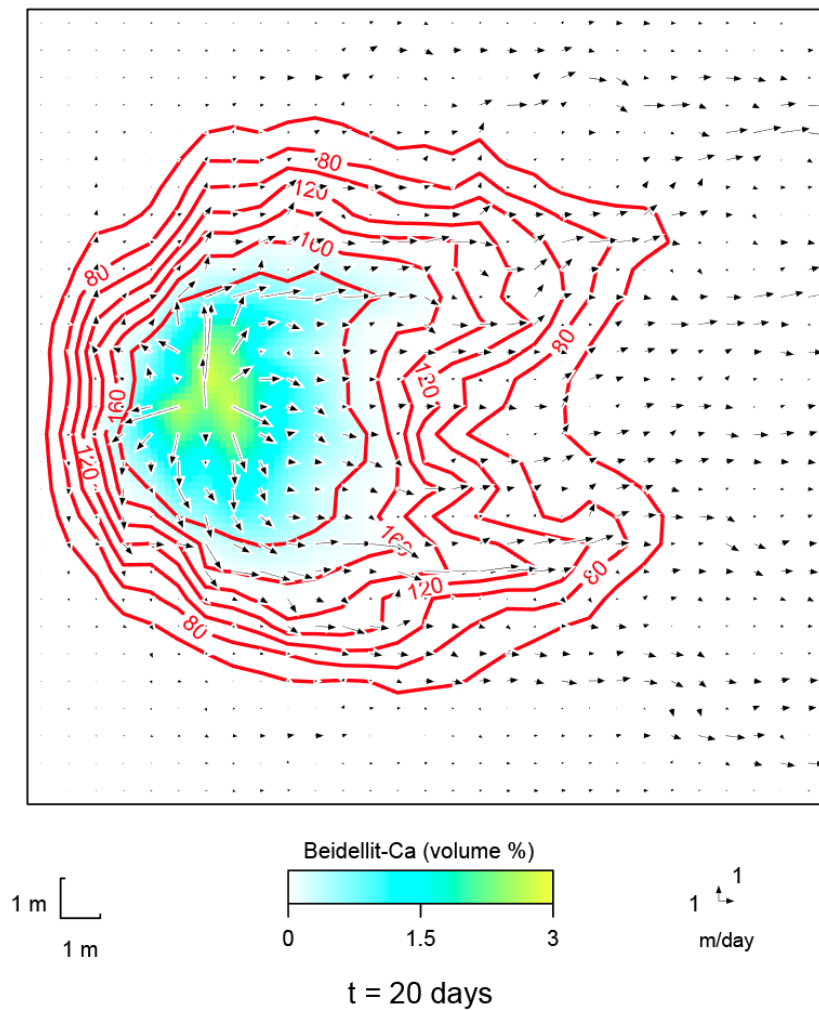


We also limit the minerals that can precipitate, as before, by going to **Config → Options...** and deselecting the precipitation option.



Trace the simulation by selecting **Run → Go**. To render the simulation results, open **Xtplot**, read the “X2t_plot_Steam2D.xtp” dataset, and select **Plot → Map View**. Then chose **Format → Contour...**, and set the “Variable type” to “Temperature”, with a contour increment of “20°C”. Click **Ok**. Next, select **Format → Color Map...**, and in the top four fields choose: “Minerals”, “Beidellit-Ca”, “volume%”, and “Linear”.

Using the various options under **Format**, including the **Quick Toggle...** dialog, you can customize your diagram. It should look like



Animate the plot to see how the steam flood proceeds through time.

4.11 X2t command line

You can start **X2t** by clicking the icon on the GWB dashboard, opening an “.x2t” file, or entering the command `x2t.exe` from the Windows “Command Prompt”.

When you start **X2t** from the command line (as opposed to clicking on the icon), you can specify a number of arguments. For example, the command

```
x2t -i my_script -d my_thermo.tdat
```

causes **X2t** to read input commands from a file “my_script”, and to use “my_thermo.tdat” as the thermodynamic database.

The following options are available from the command line:

<code>-cd</code>	Change the working directory to the directory containing the input script specified with the <code>-i</code> option.
<code>-nocd</code>	Do not change the working directory.
<code><input_script></code> <code>-i <input_script></code>	Set a file from which to read input commands.
<code>-gtd <gtdata_dir></code>	Set directory to search for thermodynamic datasets.
<code>-cond <cond_data></code>	Set the dataset for calculating electrical conductivity.
<code>-d <thermo_data></code>	Set the thermodynamic dataset.
<code>-iso <isotope_data></code>	Set the file of isotope fractionation factors to be used.
<code>-s <surface_data></code>	Set a dataset of surface sorption reactions.

Cluster Computing

Special versions of **X1t** and **X2t** are available for tracing reactive transport models on high-performance computing clusters, assemblages of computers tightly networked together to solve large problems quickly. Programs **cX1t** and **cX2t**, the cluster versions of the reactive transport models, are included in the **GWB Advanced Professional** distribution.

On a computing cluster, you run a number of copies of **cX1t** or **cX2t** at the same time, spread across the computers that make up the cluster. Each copy, or “logical process” in computing lingo, works on only a portion of the nodal blocks in the model. To solve a problem with 32,000 nodal blocks on a cluster of 32 computers, for example, you might launch 32 copies of **cX2t**, one on each computer.

Each program copy, then, would be responsible for 1,000 of the nodal blocks. As they trace a simulation, the copies periodically share data among themselves. In this way, each copy holds whatever information it needs to know about all the nodal blocks in the run, not just the blocks on which it is working.

By sharing work among computers in this fashion, the cluster can trace simulations more quickly, and trace larger simulations, than would be possible on a single computer.

5.1 Working on a cluster

Copies of programs **cX1t** or **cX2t** work together using a protocol called MPI, short for “Message Passing Interface,” that is nearly universally installed on computing clusters. An MPI installation has two parts. There is a library to which program copies can link, and through which the copies share information with each other. There is also a runtime environment that lets you launch multiple copies of a program in such a way that they work together.

By design, MPI allows you to run any number of program copies across any number of computers. In the simplest case, you might install MPI on your desktop computer. You could then run a family of program copies that work together, so long as the program is written to take advantage of MPI.

If the program in question were mono-threaded, you would likely achieve a speedup on your desktop, since the copies would run in parallel across the cores on the computer’s CPU. The **cX1t** and **cX2t** programs are multithreaded, however, which means a single copy

already makes use of all of a CPU's cores. For this reason, you would normally run just one copy of **cX1t** or **cX2t** per computer in a cluster.

In a common arrangement, you use a personal computer to access a cluster over the network. You begin by preparing input files on the personal computer, testing small jobs there before committing to the time and expense of a large simulation.

When you are ready to proceed, the cluster serves to run the full-scale simulations, either as batch processes or interactively, as described below. When a simulation completes, you retrieve the resulting "X1t_data.xtp" or "X2t_data.xtp" file from the cluster to your personal computer. There, you can render the results graphically, using **Xtplot** or the graphics program of your choice.

Note that if the cluster represents data in binary patterns that differ from those on a personal computer, you need to set the "plot = character" command in your **cX1t** or **cX2t** input, so **Xtplot** can interpret the results brought over in your ".xtp" files.

5.2 Running a simulation

Most commonly, computing clusters run a variant of the Linux operating system. We support **cX1t** and **cX2t** under Linux, as well as Windows clusters running either Microsoft's or Intel's version of MPI.

Regardless of platform, you might launch a family of MPI program copies with the `mpirun` command. For example, the command

```
mpirun -n 32 myMPIapp
```

starts a group of 32 copies of a program "myMPIapp".

Significantly, the cluster you work on may have a scheduling system for launching programs, and you might prefer to use that in place of the `mpirun` command. As well, your installation may require additional parameters on the `mpirun` command line. Please contact your system administrator for information on how to best schedule and launch MPI programs at your site.

5.2.1 Running a batch job

To run **cX1t** or **cX2t** as a batch job on a cluster, assuming you use `mpirun` as a launcher, enter a command such as

```
mpirun -n 32 cx2t -i my_input.x2t
```


A **cX2t** simulation in this case will run across 32 program copies, as given by the setting for “-n”, and each copy will be configured according to **X2t** commands in file “my_input.x2t”.

5.2.3 Interactive execution

You can also run **cX1t** and **cX2t** interactively on a computing cluster, much as you would on your desktop, within the programs’ **Command** panes. To start an interactive session, enter the command above, but without specifying an input file

```
mpiexec -n 32 cx2t
```

Lacking a source of input in this case, **cX2t** will return a prompt where you may begin entering **X2t** commands, culminating in “go” when you are ready to trigger a simulation.

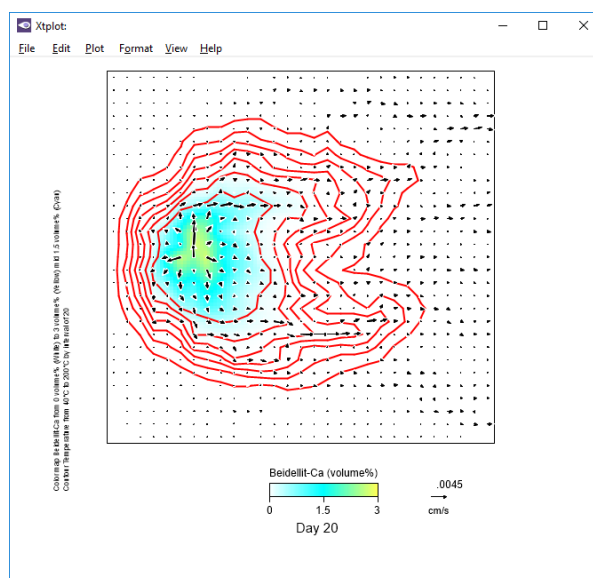
Command line processing by **cX1t** and **cX2t** is not as versatile as on the **Command** pane of the **X1t** and **X2t** GUIs. Specifically, features such as spelling completion described in the **Command Line Interface** chapter of the **GWB Reference Manual** are not available when you work interactively on a computing cluster.

Using Xtplot

Xtplot is a graphics program designed to display output from **X1t** and **X2t**. The program is similar to **Gtplot**, so if you have familiarized yourself with reaction modeling using the **GWB**, you already know much of what you need to know to run the program.

The best way to learn how **Xtplot** works is to use it! Once you have run **X1t** or **X2t** to create a “X1t_plot.xtp” or “X2t_plot.xtp” file, you start **Xtplot** by clicking on **Plot Results** on the **Results** pane. You can leave **Xtplot** active while you run further **X1t** and **X2t** simulations. Each time **X1t** or **X2t** completes a run, it signals **Xtplot** to update its display to reflect the new results.

You can also start **Xtplot** by opening any “.xtp” file, or by clicking on the **Xtplot** icon on the **GWB** dashboard. You can take input from a dataset with a different name by choosing **File → Open → Data File....** Depending on the plot type you choose, the **Xtplot** looks like



When **Xtplot** starts for the first time in a given directory, it assumes a default configuration. Upon finishing, the program saves for its next run a dataset containing the current configuration of your plots. From the **File** menu, you can specify an alternative

configuration (see [Loading and saving plot configuration](#)) or reset the entire program configuration.

Xtplot can produce various types of plots. You can generate a display of the domain in map view, showing one or several variables represented by color shading or contours. You can also display phase assemblage and predominance maps. On an xy plot, you can show how certain variables change with position or time. You can plot one or more variables versus distance across the domain, or along it. You can plot one or more variables versus time. Or, you can plot your simulation results on a variety of specialized diagrams, such as ternary diagrams and Piper diagrams. You choose the type of diagram to plot with the **Plot** pulldown menu on the menubar. The program can render the following types of variables:

- mass and volume of minerals and end members in the modeled system
- concentration, activity, and activity coefficients of dissolved species
- elemental composition of fluid, minerals, sorbate, and the bulk system
- composition of fluid, minerals, sorbate, and the bulk system, expressed in terms of thermodynamic components
- gas fugacity and partial pressure
- saturation indices Q/K with respect to various minerals and solid solutions
- bulk stable isotopic composition of the fluid, minerals, sorbate, and the entire system, as well as the composition of individual species, gases, and minerals
- fraction of various components sorbed onto mineral surfaces
- other variables such as temperature, pH, and reacted mass

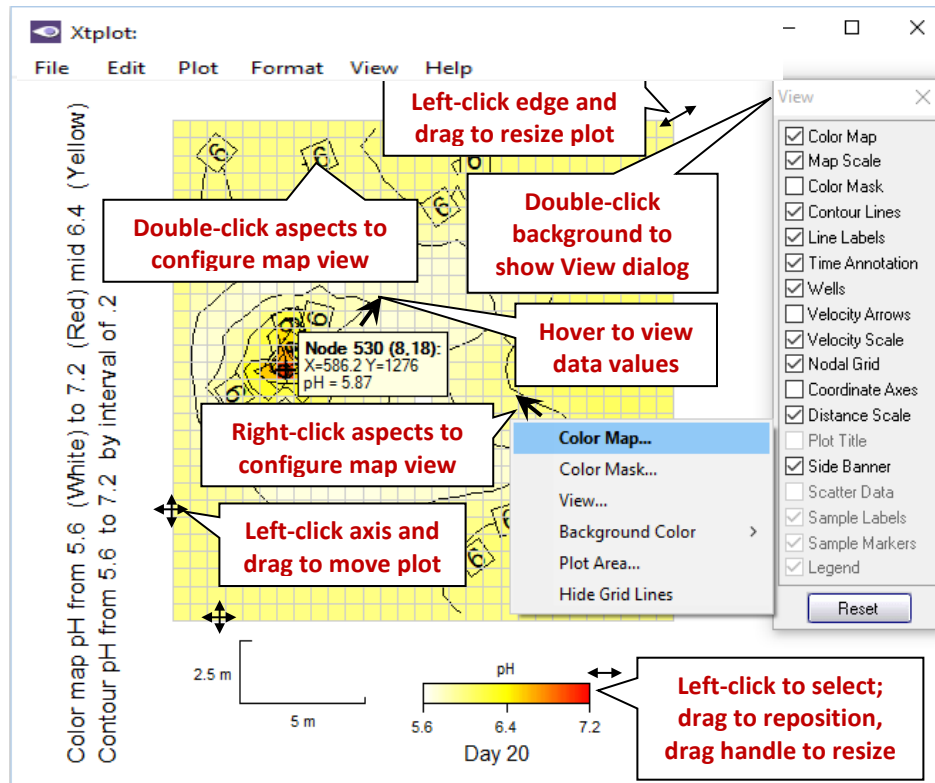
Xtplot works using information about the system modeled, including its bulk composition (i.e., its composition in terms of Na^+ , Ca^{++} , and so on), and the fluid's pH, TDS, carbonate speciation, and so on. Some of the “special” plots (Piper diagrams, ternary diagrams, and others) show variables labeled “ $\text{HCO}_3 + \text{CO}_3$ ”. Such plots show the sum of the concentrations in solution, in terms of electrical equivalents per kg fluid, of the free HCO_3^- and CO_3^{--} species. The program calculates this sum from the fluid's carbonate alkalinity, as determined by **X1t** or **X2t**. Hence, the value shown for the variable depends not only on carbonate concentration, but the fluid's pH. In contrast, plots labeled “ HCO_3 ” represent the bulk carbonate concentration of the fluid, taking no account of the speciation of carbonate to the CO_3^{--} , HCO_3^- , and $\text{CO}_2(\text{aq})$ species.

6.1 Map View

Xtplot can make plots in map view, in which variables are represented by color shading, color masking, and contour lines. You can also display a predominance diagram for any element or basis entry or render the distribution of minerals in the domain as a phase assemblage map. In addition, you may choose to show fluid velocity as represented by an arrow field, as well as the location of wells. For map view plots, the **Format** menu contains a number of options used to configure the plot. Select:

- **Quick Toggle...** to select which graphical elements of the plot are currently shown
- **Color Map...** to define a shaded color map of a variable over the domain
- **Color Mask...** to define a color mask of a variable over the domain
- **Contour...** to choose a variable to contour with lines and control how it is contoured
- **Predominance Map...** to plot the species accounting for the most mass of any original basis species or element
- **Assemblage Map...** to plot fields representing every mineral or combination of minerals throughout the domain
- **Appearance...** to open a tabbed dialog containing the following:
 - **Axes and Ticks...** to define the appearance of the coordinate axes
 - **Plot Area...** to choose the dimensions, placement, scaling ratio, and background color of the plot
 - **Title...** to define a title for the plot
 - **Font...** to control aspects of the text throughout the diagram
- **Time Step...** to choose the time step in the calculation to render on the plot
- **Animate...** to control the animation of the simulation results on your screen, or save such an animation
- **Wells...** to specify the appearance of the markers showing well positions
- **Velocity Arrows...** to control how fluid velocity is represented by an arrow field
- **Distance Scale...** to choose units for labeling distance, and to specify the position and appearance of the bar scale for distance and depth
- **Nodal Grid...** to specify the appearance of the lines showing the numerical grid
- **GSS Data...** to affect the labeling of scatter data loaded from a GSS spreadsheet and the related legend
- **Side Banner...** to specify the appearance and placement of the informational banner along the side of the plot

Alternatively, you can interactively modify many aspects of the plot, including lines, labels, axes, scales, velocity arrows, and grid lines. A **right-click** on an aspect displays a menu showing the options available. A **double-click** on an aspect brings up a related dialog. **Left-click** on an aspect to select it for modification. An aspect has been selected when its color changes, as in the case of lines, or when it is surrounded by manipulation handles, in the case of text, markers, and scales. Change the size of text, markers, scales, and the plot itself by dragging the sizing cursors. Adjust the placement of the plot by clicking within the plot or on either axis and dragging it. The plot title can be moved in the same way.



6.2 XY Plot configuration

For xy plots, you can plot your calculation results against X position (distance along the domain), Y position (distance across the domain), or time.

Choose **Plot** → **XY Plot...** to specify the configuration of the xy plot. As with the map view, you can interactively modify many aspects of the plot. A **double-click** on an aspect brings up the **XY Plot Configuration** dialog. A **right-click** on an aspect displays a menu showing the options available. **Left-click** on an aspect to select it for modification. Dragging an inner- or outermost axis tick mark, number, or grid line will shift or scale, respectively, the axis range (see **Using Gtplot** in the *GWB Essentials Guide*).

The **X Axis** tab lets you select the independent variable to appear on the *x* axis and specify the way it is plotted. Use the **Display** menu to choose between:

- **X position** to plot versus position along the *x* direction (the default)
- **Y position** to plot versus position along the *y* direction
- **Time elapsed** to plot against relative time
- **Time and date** to plot against an absolute date-time variable, when available
- **Time of day** to plot against a 24 hour cycle, when available

The **Y Axis** tab lets you select the variables to appear on the y axis and specify the way they are plotted. Use the **Variable type** menu to control the type of variables to appear along the y axis. Choose from the variable type options:

- **Chemical parameters** such as temperature, pH, and mass solution
- **Physical parameters** such as fluid density, porosity, and fluid velocity
- **Reactant properties** such as net reaction progress and reaction rates
- **Components** in the fluid, minerals, sorbate, and the bulk system (fluid plus sorbate and rock)
- **Component Kd's** of the various components that may be sorbed onto mineral surfaces
- **Sorbed fractions** of the various components that may be sorbed onto mineral surfaces
- **Surface parameters** such as charge density, potential, and sorbing surface area
- **Species** to plot either the concentration, activity, or activity coefficient of one or more aqueous species
- **Minerals** to represent the masses or volumes of minerals and solid solution end members over the reaction path
- **Mineral saturation** to plot the saturation indices (Q/K) of the fluid with respect to one or more minerals and any solid solutions considered
- **Gas fugacity** or **Gas partial pressure** of one or more gases in the fluid
- **Elemental composition** of either the fluid, the rock, the sorbate (i.e., surface complexes), or the system (fluid plus sorbate and rock)
- **Isotopic composition** of the bulk fluid, rock, sorbate, and system, as well as individual species, for active isotope systems

Use the **Filter** menu to set, for species or mineral saturation plots, a basis species to consider. In a typical reaction path calculation, so many dissolved species are considered, and saturation states are determined for so many minerals, that the data fit poorly on a single plot. For this reason, **Xtplot** can group species and minerals into sets of just those that contain a given basis species. You can choose to plot, for example, the species containing Na^+ , or the saturation indices of minerals containing Al^{+++} . Alternatively, you can choose to consider all species and minerals.

Use the **Display** menu to set whether one or more variables are shown on the plot. The default setting, **several values at one time** (or **several values at one node**, if time is chosen for the x axis) plots one or more related variables (e.g., the volume fractions of each of the minerals in the system) at a single time level (or a single node). The alternate setting, **one value at several times** (or **one value at several nodes**, if time is chosen for the x axis) plots at one or more time levels (or nodes), a single variable (e.g., the volume fraction of one mineral).

The selection of a variable type displays a list of possible variables. Click on a variable you wish to select. For plots of several variables versus distance or time, you can select

more than one entry from the list. Use **Ctrl+click** to select multiple variables. Use **Shift+click** to select a range of variables. In the case of a plot of chemical, physical, reactant, or surface parameters, you can group only variables with similar units on the plot. Use the **First**, **Previous**, **Next**, and **Last** buttons to cycle through the variables.

Use the following to specify the way the variables are plotted:

- **Auto-scale** to set the data range for the axis to span the data to be plotted
- **Minimum**, **Maximum**, and **Tick factor** to set the data range for the axis
- **Reverse axis** to reverse the sense of the axis
- **Units** to select alternative units, if any, for the axis
- **As** to plot values as elemental equivalents, or as protonated or deprotonated species equivalents
- **Type** to set the axis to a linear, log, or delta scale. A delta scale shows change in a variable's value from the initial point in the reaction path. This latter option is available only on plots versus time

The **Position** tab lets you select the position in the domain for which data should be rendered. Position may refer to the node column or row, or (in plots versus time) individual nodes.

The **Time Level** tab lets you select, on plots versus distance, the time level or levels for which data should be rendered. If there is more than one time level step, you may choose **Animate...** to cycle through them automatically.

Use the **First**, **Previous**, **Next**, and **Last** buttons to cycle the plot through the positions or time levels on the related tabs.

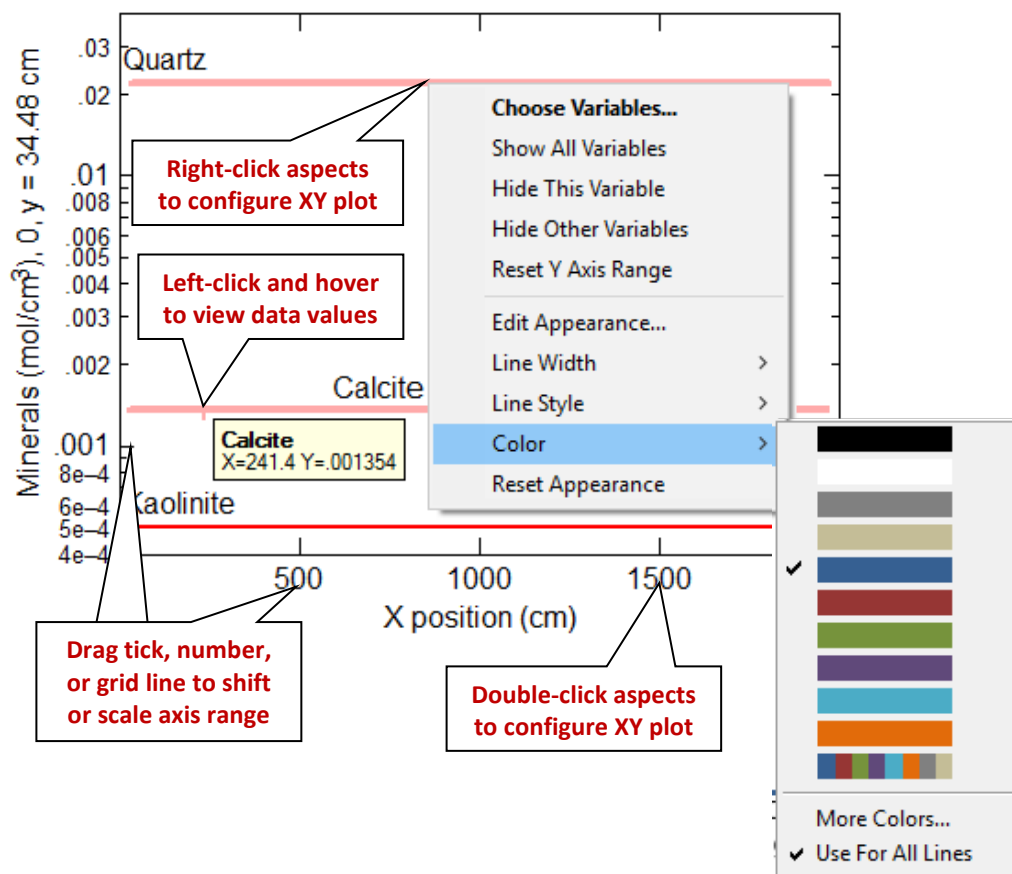
In plots versus distance, you can make the **Display** selection from either the **Y Axis** or **Time Level** pane. In plots versus time, you can make the **Display** selection from either the **Y Axis** or **Position** pane. Choosing **one value at several times** automatically selects six equally-spaced time levels, if selections have not been made previously. At any time, you can hit **Reset** on the **Time Level** or **Position** pane, then choose **one value at several times** (or **one value at several nodes**), then hit **Apply**, to automatically select multiple equally-spaced times (or nodes).

6.3 Plot types

In addition to the map view and xy plot, you can plot the results of the **X1t** or **X2t** calculations on any of the special plot types. These include ternary diagram, Piper diagram, Durov diagram, Schoeller diagram, Stiff diagram, radial plot, bar chart, and pie chart. Use the **Plot** menu to control the type of plot produced and the configuration of the plot. Each selection invokes a dialog box specific to the type of plot in question (see **Using Gtplot** in the *GWB Essentials Guide*).

6.4 Editing plot appearance

As with the map view and xy plots, you can interactively modify many aspects of any of the plot types. The details of using active items, aspect menus and dialogs to edit the plot appearance are given in the **Using Gtplot** section of the *GWB Essentials Guide*.



6.5 Scatter data

Xtplot can overlay the data in a **GSS** data sheet (a .gss file) as “scatter data” on a plot of the results of an **X1t** or **X2t** calculation. The program can add scatter data to any of the plots it makes, except bar and pie charts. To read a .gss file into **Xtplot**, select **File** → **Open** → **Scatter Data...**. Clear scatter data from a diagram on the **Open** → **Scatter Data...** dialog, with the **OFF** button.

The **Scatter data** section of the **Using Gtplot** chapter in the *GWB Essentials Guide* gives details on plotting scatter data from .gss files in **Gtplot**. Follow the information in that section, as **Xtplot** works in much the same manner as **Gtplot**.

In **Xtplot**, xy plots appear with either “X position”, “Y position”, or “Time elapsed” on the *x* axis, as well as “Time and date” or “Time of day”, when available. To add scatter data to an xy plot, then, you need to include in your .gss file the chemical or physical parameter corresponding to the *x* axis variable: “X position”, “Y position”, “Time elapsed”, “Date”, or “Time of day”. To plot scatter data on a map view plot, you must include sample values for both “X position” and “Y position”.

By default **Xtplot** checks a .gss file for the *x* and *y* axis variables only. When a .gss file includes additional time or position information, however, **Xtplot** can optionally restrict the display of scatter data. In a plot of concentration vs. time, for example, you may wish to display scatter data only at the node corresponding to the sampling point. If the .gss file includes this position, check “Plot only at specified times / positions” to limit scatter data to the closest node. If unchecked, scatter data would plot with results from any position.

To coordinate the plotting of sample dates and times in the **GSS** data sheet on the results of the calculation, set an explicit starting date and time in **X1t** or **X2t**.

In **GWB** releases 7.0 and earlier, the program took scatter data from a specially formatted text file, rather than a .gss data sheet. Legacy scatter files are still supported and are described in the **Scatter Data** chapter of the *GWB Reference Manual*.

6.6 Loading and saving plot configuration

Upon finishing, **Xtplot** writes into the user’s working directory a file, “xtplot_conf.xtc”, containing the configuration of the current plots. When the program starts again in the same directory, it reads the file and assumes the same configurations.

Choosing **File → Reset Configuration** or the `reset` option from the command line (see [Xtplot command line](#)) returns the configuration for each plot type to its default state.

You can save the settings for several types of plots in different configuration files. To do so, select **File → Save As...**, or use MS Windows to copy the “xtplot_conf.xtc” file of interest to another filename. You may then specify that file as the configuration for a later **Xtplot** run from the command line (the `-c` flag; see [Xtplot command line](#)) or read it into **Xtplot** by selecting **File → Open → Configuration...**

Exiting the program by choosing **File → Abort (No Save)** causes an immediate exit from the program; the plot configuration is lost.

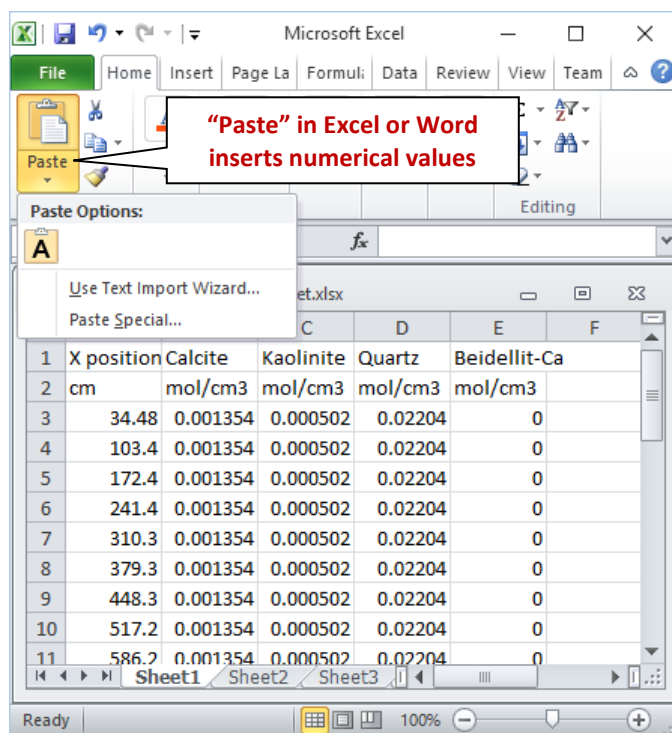
6.7 Exporting the plot

Xtplot makes it convenient to use the plots you create in articles, reports, presentations, and databases. You can copy the current plot to the clipboard and then paste it into a variety of applications in a format meaningful to the application.

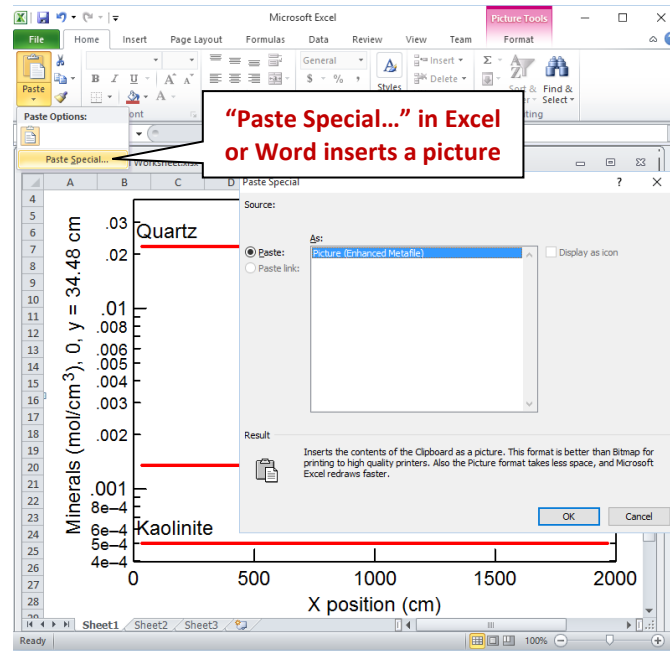
To copy a plot, use **Edit → Copy** or **Ctrl+C**. If you paste the plot into MS PowerPoint, it will appear as an EMF (an MS Enhanced Metafile) graphic object. Pasting into Adobe Illustrator places a native AI graphic.

If you paste the plot into MS Excel or a text editor, such as Notepad or MS Word, the numerical values of the data points that make up the lines on the plot will appear in spreadsheet format.

You can control the format in which the plot is copied to the clipboard by selecting **Edit → Copy As**. You can choose to copy the plot as an AI object, an EMF object, or a bitmap, or copy the data points in the plot as tab delimited or space delimited text. Use the tab delimited option to paste the data into a spreadsheet program like MS Excel. For examining the data in a text file created with an editor like Notepad or MS Word, the space delimited option writes a nicely aligned table.



In MS Word or MS Excel, use **Paste Special...** to paste the plot as a picture instead.



Use **File → Save Image...** to copy the plot, in your choice of formats (selected under **Save as type:**), from the graphics area to a file in the specified directory. The **Enhanced Metafile** option, for example, saves the plot image in a format that can be read by most art and illustration programs (see **Graphics Output** in the *GWB Reference Manual*). The other file formats available are: **PNG, JPEG, TIFF, Bitmap, Adobe Illustrator, PDF, Scalable Vector Graphics, Compressed SVG, Encapsulated PostScript, Color PostScript, and B/W PostScript**. When saving a PNG, JPEG, TIFF, or bitmap file, you may specify the quality of the saved image by choosing its resolution: High, Medium, Low, or Custom. Use **Custom...** to set the pixel width and height of the image, and to choose whether to preserve the aspect ratio of the plot. Use the **Spreadsheet File (Tab delimited)** or **Text File (Space delimited)** option to save into a table the numerical coordinates of the data points on the plot. The spreadsheet table may be read directly into many popular spreadsheet programs.

Certain graphics types support font embedding. PDF files should always display and print properly, regardless of fonts installed on the system. PostScript files should also, if you have used the option to embed fonts. If you may want to edit the PostScript file, however, you should deselect the option to embed fonts, because programs such as Adobe Illustrator may restrict your ability to edit a document using embedded fonts. To edit these files, be sure that all of the required fonts are installed on your computer (see **Font for data markers** in the *GWB Reference Manual*).

When importing AI graphics to Adobe Illustrator, the program may prompt you to update the legacy text before you can edit the file. In this case, choose “Update”. You need to release the clipping mask before you attempt to edit individual elements of the plot. Use the “Ungroup” and “Group” functions when repositioning or modifying elements.

6.8 Xtplot command line

To run **Xtplot**, click on the **Xtplot** icon on the GWB dashboard, or open an “.xtp” or “.xtc” file. The program can also be initiated from the Windows command prompt by typing `xtplot.exe`. Starting the program in this way allows you to make use of the command line arguments described below.

Xtplot accepts a number of arguments from the command line. For example, the command

```
xtplot -i X1t_plot1.xtp -c Config1.xtc
```

causes **Xtplot** to read as input the file “X1t_plot1.xtp”, and to use the plot configuration stored in “Config1.xtc”.

The following options are available from the command line:

<code><input_data></code> <code>-i <input_data></code>	Set the “X1t_plot.xtp” or “X2t_plot.xtp” dataset, produced by an X1t or X2t run, which contains the data to be plotted. The program, by default, looks for file X1t_plot.xtp in the user’s working directory.
<code>-c <config_file></code>	Set the configuration file to be read at startup. By default, the program reads the file <code>xtplot_conf.xtc</code> if it exists in the working directory.
<code>-scat <scatter file></code>	Take scatter data from the named dataset. By default, Xtplot does not plot scatter data.
<code>-graph <plot_type></code>	Set the type of plot displayed at startup. By default, it is set from the configuration file. Choose from: map, xyplot, ternary, piper, durov, schoeller, stiff, radial, bar, and pie.
<code>-reset</code>	Set the default configuration at startup; do not read <code>xtplot_conf.xtc</code> .

Table 6.1	Keyboard shortcuts in Xtplot
Ctrl+Shift+A	Copy plot to the clipboard as an Adobe Illustrator file
Ctrl+Shift+B	Copy plot to the clipboard as a bitmap
Ctrl+C	Copy plot to the clipboard
Ctrl+Shift+E	Copy plot to the clipboard as an Enhanced Metafile
Ctrl+F	Refresh display
Ctrl+Shift+G	Get scatter data
Ctrl+L	Load a configuration file
Ctrl+M	Save the graphic image to a file
Ctrl+O	Open an input (“X1t_plot.xtp” or “X2t_plot.xtp”) file
Ctrl+P	Print the plot
Ctrl+Q	Quit Xtplot, save configuration
Ctrl+Shift+Q	Abort Xtplot, do not save configuration
Ctrl+R	Reset the plot configuration
Ctrl+S	Save the current configuration to a file
Ctrl+Shift+S	Copy plot to the clipboard as Spreadsheet (tab delimited)
Ctrl+Shift+T	Copy plot to the clipboard as Text (space delimited)
Ctrl+U	Update the plot from the current input file
Ctrl+X	Reset data range on the x axis
Ctrl+Y	Reset data range on the y axis
Ctrl+Z	Reset data ranges on the x and y axes
F1	Open GWB Reactive Transport Modeling Guide

Appendix: Heterogeneity

When running an **X1t** or **X2t** simulation, you may wish to set a property, such as permeability or the initial fluid chemistry, so that it varies with position in the domain. You may specify such heterogeneous properties for a variety of variables in the simulation, in various ways.

You can set heterogeneous values for the following variables:

- The chemical composition of the initial fluid, including pH, concentrations of the basis entries (e.g., Na^+ , Ca^{++}), gas fugacities, mineral masses, and so on.
- The groundwater flow field and initial temperature within the domain.
- The amounts of reactant species and their cutoff values, the final value for sliding activity and fugacity reactants, and the initial biomass for microbial reactants.
- For kinetic reactants: the rate constant, activation energy, pre-exponential factor, specific surface area, nucleation area, and critical saturation index.
- The properties of the medium, including its porosity, permeability, transverse and lateral dispersivity, thermal conductivity, internal heat source, and ion exchange capacity.
- In dual porosity models, the volume fraction of the stagnant portion of the medium; its porosity, diffusion coefficient, retardation factor, and thermal conductivity; and the half width and diffusion length of the individual stagnant blocks or spheres.
- The lengths (Δx) and widths (Δy) of nodal blocks.
- The mobility of a complexing surface—the fraction of the surface that can move in the model by advection and dispersion.

Each variable that can be set as a heterogeneous value is known as a field variable, and these variables are identified in the **X1t Commands** and **X2t Commands** sections of the ***GWB Command Reference*** by the command argument `<field_variable>`.

The programs at the onset of the run evaluate field variables for each node in the domain. A number of properties—the internal heat source or the rate constants for kinetic reactions, for example—can be set to change with time. The programs evaluate such transient field variables not only initially, but upon undertaking each time step over the course of a run.

The **X1t Commands** and **X2t Commands** sections of the *GWB Command Reference* identify the properties that can be set to vary in time. To specify how a field variable behaves, you check or uncheck the ☒ “transient” box in the GUI next to where the variable is set. Alternatively, append on the command line the keyword `transient` or `steady` when defining the variable. A field variable, by default, holds steady.

You can set a field variable in various ways:

- As a simple value, in which case the property is homogeneous. By entering a `?`, you reset the variable to its default state.
- As a value \pm a standard deviation, in which case the property is a random field with a Gaussian distribution.
- As values entered into a table within an external dataset.
- As values within an “in-line table” that appears directly in the input data stream, rather than in an external dataset.
- As the result of evaluating an expression, entered as a character string.
- As the result of evaluating a basic script you supply.
- As the result of evaluating a compiled function written in C/C++.

The last three options above are very similar in implementation to the method for specifying custom rate laws for kinetic reactions, as described in the **Custom Rate Laws** section of the *GWB Reaction Modeling Guide*. Before using one of these options, you should read that section.

A.1 Values and random fields

You set a uniform value for a field variable by assigning a value, just as you would any other variable. For example, the commands

```
porosity = 0.30
porosity = 30%
```

set the initial medium porosity to a homogeneous value of 30%.

To set a variable to a random field, you define a standard deviation using the `std_dev` keyword. For example, the commands

```
porosity = .25, std_dev = .05
porosity = .25 +/- .05
```

set porosity to vary about a mean value of .25 (25%) in a Gaussian distribution with a standard deviation of .05.

You can specify the seed for the random number generator with the `seed` keyword.

```
porosity = .25 +/- .05 seed = 999
```

By default, the standard deviation is 0 (porosity is constant over the domain) and the seed is 0.

The programs try to protect against certain nonsense values, such as negative porosities, by limiting the allowable range of values they assign.

A.2 Table datasets

You can define a field variable by assigning values node-by-node from a table dataset. For example, you could type the command

```
porosity = my_table.txt
```

to define a heterogeneous porosity field using values stored in the dataset “my_table.txt”. Note that table datasets must end with the extension “.txt”.

The program assigns values by scanning across the domain. It begins by taking values from the first line in the table and assigning them to nodes along the bottom-most row in the domain, left to right. If there are more nodes than values in the first line, the program returns to the beginning of the line once it reaches the end and continues scanning. If there are more values on the line than nodes, the excess values are discarded.

The program continues by assigning values from the second line of the table to the overlying row of nodes. (Since the **X2t** domain is indexed from bottom to top, the table dataset is inverted with respect to the domain.) The program next assigns the third line to the third row of nodes, and so on. If the program reaches the end of the table dataset before the top line of nodes, it returns to the top of the dataset. If the dataset contains more lines than there are rows of nodes, the unread lines are ignored.

Table datasets may contain comments. Comment lines begin with a number sign (“#”) and may appear at any point in the table. You can use the star symbol (“*”) to repeat values in the table. For example, “3*0” is the same as “0 0 0”.

A.3 In-line tables

Rather than storing a table in an external dataset, you can set values at each node in the domain directly within your input data. To do so, set the field variable in question to a table of data enclosed by curly brackets (“{” and “}”).

The data appear with rows on separate lines, in the same format as an external table dataset, as described in the previous section. For example, you could type the command

```
porosity = {
.10 .15 .20
.15 .20 .25
.15 .20 .20
}
```

to define a heterogeneous porosity field, using the values shown. In practice, of course, the table would likely contain many more rows and columns.

As an alternative to setting rows on separate lines, you can use the vertical bar character (“|”) as a separator. The above command, then, can equivalently appear as

```
porosity = { .10 .15 .20 | .15 .20 .25 | .15 .20 .20 }
```

In either case, the program assigns values by scanning across the domain, in the manner described in the previous section for table datasets. As with table datasets, you can use the star symbol (“*”) to repeat values. The command

```
porosity = { .10 .15 .20 97*.25 }
```

sets porosity in a domain of 100 blocks or less to 25%, except for the left-most three nodes, or columns of nodes.

A.4 Simple expressions

If you can describe variation in the field variable with an equation, you can simply type in the form of the equation. **X1t** or **X2t** will evaluate the equation and use the result to assign the property in question.

You specify the expression to be evaluated with the `equation` or `eqn` keyword. For example, you might enter the command

```
porosity eqn = \
"0.3*SIN(4*Xposition/Length)*COS(4*Yposition/Width)"
```

This command will cause porosity to vary periodically over the domain.

Table A.1 lists parameters describing the current nodal block that you can use when setting heterogeneity in this manner.

Table A.1 Internal parameters describing the current nodal block, available for setting heterogeneous properties. Parameters are case sensitive.

Temperature	
temperature, TC, Tempc	Temperature (°C)
TK	Absolute temperature (K)
Domain	
Nnode	Number of nodal blocks in total
Nx	Number of nodal blocks along the x direction
Ny	Number of nodal blocks along the y direction
Length	Size of domain along the x direction (cm)
Width	Size of domain along the y direction (cm)
Height	Size of domain along the z direction (cm)
Radius1	Beginning radius of radial or spherical domain (cm)
Radius2	Ending radius of radial or spherical domain (cm)
Wedge_angle	Radial angle of radial or spherical domain (radians)
Indexing	
Inode	Node index
Ix, Xindex	Index of nodal block along the x direction
Jy, Yindex	Index of nodal block along the y direction
Nodal block	
Xposition	Position of center of nodal block along the x direction (cm).
Yposition	Position of center of nodal block along the y direction (cm).
Deltax	Length of nodal block (cm).
Deltay	Width of nodal block (cm).
Deltaz	Height of nodal block (cm).
bulk_volume, Vbulk	Bulk volume of nodal block (cm ³).
inert_volume, Vinert	Non-reacting volume in nodal block (cm ³).
Xfree	Free-flowing fraction of nodal block.
freeflowing, Vfree	Volume of free-flowing zone in block (cm ³).
stagnant, Vstag	Volume of stagnant zone in block (cm ³).
Xarea	Area (cm ²) of nodal block normal to x direction
Yarea	Area (cm ²) of nodal block normal to y direction
Poros0	Sediment initial porosity, as a fraction
porosity	Sediment porosity, as a fraction

Time marching

Time	Time (s)
Tstart	Starting time (s)
Tend	Ending time (s)
Deltat	Length of time step (s)
Theta	Time weighting variable
Xi	Reaction progress
PV	Pore volumes displaced

A.5 Basic scripts

You can define heterogeneity in terms of more sophisticated algorithms by preparing a script in a simplified form of the BASIC language, as described under **Custom Rate Laws** in the *GWB Reaction Modeling Guide*, and save it in a file. The name of the script file must end with the extension “.bas”.

You specify the name of the file by assigning it to the property in question. For example, the command

```
porosity = Poros_script.bas
```

tells **X1t** or **X2t** that it should evaluate the script in the specified file to figure initial porosity. The program looks for this file first in the current working directory, and then in the directory where the GWB is installed (for example, “C:\Program Files\GWB”). You can, of course, specify a full path name to the file to avoid ambiguity.

You can alternatively set your script in-line within your **X1t** or **X2t** command input. The commands

```
porosity = script {
    xfrac = Xposition/Length
    IF xfrac < 0.5 THEN por = 0.3 ELSE por = 0.2
    RETURN por
}
```

set out your BASIC script directly within your input file, without need to refer to an external file. The script itself is the portion of the command between the brackets.

Your script can access the parameters in [Table A.1](#). Parameters in BASIC scripts are copies of the values carried internally by the program; these values cannot be changed from your script.

An example of a script for assigning a heterogeneous value is

```
xfrac = Xposition/Length
yfrac = Yposition/Width

IF xfrac > 0.4 AND xfrac < 0.6 THEN 20 ELSE 40
20:
    IF yfrac > 0.4 AND yfrac < 0.6 THEN p=0.5 ELSE p=0.25
    RETURN p
40:
    RETURN 0.25
```

In this example, the script returns a porosity of 25%, except for the area in the center of the domain, where porosity is set to 50%.

A.6 Compiled functions

For maximum flexibility, you can write your own function in C++ that returns the value to be assigned at a node to the field variable. When **X1t** or **X2t** runs, the program will link to your function and call it directly.

There are three steps in the procedure:

1. Write a function that returns the property's value, using the "gwb_context.h" and "node_param.h" header files supplied with the GWB software release. Copies of these files are found in the "src" subdirectory where the GWB is installed.
2. Compile the function and link it along with any other functions you may have written into a dynamic link library (or DLL). The DLL must end in the extension ".dll".
3. In your **X1t** or **X2t** input, specify the names of the library and function, separated by a colon (":"). For example, the command

```
porosity = my_lib.dll:poros
```

tells **X1t** or **X2t** to look for a function "poros" in library "my_lib.dll". As with script files, **X1t** or **X2t** looks for the library first in the current working directory, then where the GWB is installed.

Each function takes three parameters:

1. The index i of the node in question. For a node in column ix and row jy , the index i equals $jy \times N_x + ix$, where N_x is the number of nodes along x .

2. Reference to a “Nodal_block” data structure that describes the nodal block in question. This reference should be named “n”. The form of this data structure is defined in a header file “gwb_context.h”, which is distributed with the GWB.
3. Reference to a “GWBcontext” data structure that contains information on the domain and nodal block properties (domain length, nodal block position, and so on) in the **X1t** or **X2t** simulation. This reference is named “c”. Again, the form of this data structure is defined in “gwb_context.h”.

The parameters in [Table A.1](#) are available to your function. These parameters are defined in a second header file “node_param.h”, which you should include at the top of the function. You can also use as parameters any of the entries in the “GWBcontext” data structure, as set out in “gwb_context.h”. To debug functions, you can use the `printf` function (but not `cout`) to display data.

As an example of such a function, we take the example of setting porosity from the previous section. The function, named `poros`, might be written

```
#define EXPORT extern "C" __declspec(dllexport)
#include <windows.h>
#include <stdio.h>
#include <math.h>
#include "gwb_context.h"

EXPORT double poros(int i, Nodal_block &n, GWBcontext &c)
{
    #include "node_param.h"

    double xfrac = Xposition/Length;
    double yfrac = Yposition/Width;

    if (xfrac > 0.4 && xfrac < 0.6 &&
        yfrac > 0.4 && yfrac < 0.6)
        return 0.5;
    else
        return 0.25;
}
```

For further explanation of compiled functions and how to link them into a DLL, see **Custom Rate Laws** in the *GWB Reaction Guide*.

Appendix: Importing from MODFLOW

You can import into **X2t** a flow field calculated with the **MODFLOW** groundwater flow model, rather than having **X2t** determine the flow field itself. **MODFLOW** is a widely used computer program originally developed by the U.S. Geological Survey. It is available as public domain software that can be downloaded over the Internet, as well as in various proprietary versions. You must use **MODFLOW-2000** or **MODFLOW-2005**; **X2t** does not work with older releases.

To import a flow field, set up a groundwater flow simulation in **MODFLOW** according to the following guidelines:

- The **MODFLOW** domain should consist of a single layer. If it includes more than one layer, **X2t** will pick up only the uppermost.
- We currently recommend setting the nodal block dimensions Δx and Δy to be invariant over the domain.
- The model can incorporate any of the standard **MODFLOW** packages, including injecting and producing wells, areal recharge and discharge, flow to and from streams, and evapotranspiration.
- You can set up in **MODFLOW** either a steady-state or transient simulation.
- Be sure to specify preferred units for time and length; do not use “undefined” for these settings.
- **X2t** expects to find a volume balance for groundwater flow, well injection and production, recharge, etc. at each nodal block. If a close balance has not been achieved, **X2t** will issue a warning message.
- In the Output Control file, set the option to “save budget” for each of the time steps in your simulation for which you generate output.
- You need to set certain options for output to **MODFLOW**’s budget file, as described below.

B.1 Discretization and budget files

When you run **MODFLOW**, you supply various input files, including a discretization file. When it runs, the program can write, among other datasets, a budget file. You should give the discretization file the suffix “.dis”, and the budget file, “.bud”.

The discretization file contains information about the finite difference domain and the time stepping in the **MODFLOW** model. The budget file is a binary dataset containing information about the flow field. In setting up your **MODFLOW** simulation, observe the following guidelines:

- The budget file needs to contain data describing flow among the nodal blocks. Specify the unit identifier for the budget file as the value for the “IxxxCB” variable, where “xxx” is the flow package in use. The variable, for example, might be “IBCFCB”, “ILPFCB”, and so on.
- The budget file also should contain data for nodal block flows resulting from the external stresses, such as wells, recharge, evapotranspiration, and so on in the simulation.

To set **MODFLOW** to write these data, you specify in the input file for the package in question the unit identifier for the budget file. For a package “xxx”, set variable “IxxxCB” (e.g., IWELCB, IRCHCB, IETSCB, etc.) to the unit identifier.

B.2 Importing to and running X2t

You run **X2t** by setting up the initial system and boundary fluids, as usual. Rather than setting the domain size and discretization explicitly, however, you let **X2t** derive this information from your **MODFLOW** files.

Once you have run **MODFLOW** to produce a budget file, select the **Flow** pane in **X2t**. Click on the “imported from MODFLOW” option, then click on “Browse...” and select the discretization file. The program will expect to find a budget file with the same root name. For example, if your discretization file is named “myrun.dis”, **X2t** will try to open “myrun.bud”. Alternatively, in the **Command** pane enter the command

```
modflow "myrun.dis"
```

The command `show modflow` displays the current file settings.

X2t will scan the files and set up its domain, including length and width, number of nodal blocks along *x* and *y*, well placement, and so on. For transient runs, **X2t** will also set the beginning and ending time for the simulation. Finally, it will import the **MODFLOW** flow fields for later use.

Once you have read the **MODFLOW** files, you should avoid executing commands that would alter the domain settings. For example, if you read a “.x2t” file saved from a

previous run, the embedded commands `Nx`, `Ny`, `length`, and so on will overwrite any settings that you have previously imported from the **MODFLOW** simulation. If you wish to alter the domain size or discretization, or the time span of the run, you should do so in the **MODFLOW** input files, re-run **MODFLOW**, and import anew the discretization and budget files.

When you run **X2t**, the program will use the composition of whatever fluid is associated with a well to account for inflow from that well to the domain. Other inflows, such as areal recharge and base flow from streams, as well as inflows at constant-head nodal blocks, will be taken to have the composition whatever fluid is currently active on the left boundary.

B.3 MODFLOW log file

When you import a **MODFLOW** flow field in an **X2t** simulation, **X2t** writes a log file containing debugging information. The file is named “mf_log.txt” and is located in the user’s profile directory (found by typing %appdata% in the Windows Explorer Address bar, e.g., “C:\Documents and Settings\userName\Application Data\GWB”).

The log file contains information about the time levels imported from the **MODFLOW** budget file. Optionally, the file includes volume imbalances for nodal blocks where imbalances exceeding .01% are encountered, as well as the transition points in the **X2t** simulation (i.e., the times when **X2t** starts using flow data from a new step). To include this optional information, set the `long` option on the `modflow` command.

B.4 Examples

Four examples of input files for importing **MODFLOW** results to **X2t** are included in the GWB Professional distribution and installed automatically with the GWB in the “Script” directory, under “Modflow” (e.g., in “\Program Files\GWB\Script\Modflow”).

The first example is based on the “str” test file distributed with the **MODFLOW-2000** release. Files “Example1.ba6”, “Example2.bc6”, and so on are input files for running **MODFLOW**. “Example1.bud” is the resulting budget file, and “Example1.x2t” is a file that invokes **X2t**, reading the flow field from the **MODFLOW** files.

This example is a single-layer model that invokes the **MODFLOW**’s stream package module. The following modifications to the **MODFLOW** test file were made to allow **X2t** to read the results:

- The filenames in “Example1.nam” were altered to reflect the new path and file names.
- The length unit in the discretization file (“Example1.dis”) was changed from undefined to feet.

- A unit identifier and filename for the budget file (“Example1.bud”) were appended to the name file (“Example1.nam”).
- The output control file “Example1.oc” was set to save budget data for node-by-node flow after each step.
- The variable IBCFCB in the block centered flow package (“example1.bc6”) was set to 50, the unit number of the budget file.
- The variable ISTCB1 in the stream package (“Example1.str”) was set to 50, the unit number of the budget file.

The second example is based on the “retest” test model distributed with the **MODFLOW-2000** release. In this example, a single layer of nodal blocks is used to simulate aquifer-reservoir interaction. The following modifications to the **MODFLOW** input files were made to allow **X2t** to import the model correctly:

- The filenames in “Example2.nam” were altered.
- The length unit in the discretization file (“Example2.dis”) was changed from undefined to feet.
- A unit identifier and filename for the budget file (“Example1.bud”) were appended to the name file (“Example2.nam”).
- The output control file “Example2.oc” was set to save budget data for node-by-node flow after each step.
- The IBCFCB variable in the block-centered flow package (“Example2.bc6”) was set to 50, the unit number of the budget file.
- The IRESCB variable in the reservoir package (“Example2.res”) was set to 50.
- The IGHBCB variable in the general-head boundary package was set to 50.

The third example is based on the “etsdrt” test model distributed with the **MODFLOW-2000** release. In this example, a single layer of nodal blocks is used in a model that includes flow due to the recharge, drain return, and evapotranspiration packages. The following modifications to the **MODFLOW** input files were made to allow **X2t** to import the model correctly:

- The filenames in “Example3.nam” were altered.
- A unit identifier and filename for the budget file (“Example3.bud”) were appended to the name file (“Example3.nam”).
- The output control file “Example3.oc” was set to save budget data for node-by-node flow after each step.
- The ILPFCB variable in the layer-property flow package (“Example3.lpf”) was set to 50, the unit number of the budget file.
- The IDRTCB variable in the drain-return package (“Example3.drt”) was set to 50.
- The IETSCB variable in the evapotranspiration segments package (“Example3.ets”) was set to 50.
- The IRCHCB variable in the recharge package (“Example3.rch”) was set to 50.

The fourth example, like the first, is based on the “str” test model like Example1, but the model includes wells. The following modifications to the **MODFLOW** input files were made to allow **X2t** to import the model correctly.

- The modifications from the first example are repeated.
- File “Example4.wel”, which invokes the well package, was created and added to the model.
- The IWELCB variable in the wells package (“Example4.wel”) was set to 50, the unit number of the budget file.
- The well package file information was added to the name file (“Example4.nam”).

Appendix: Further Reading

The following literature references, from the many hundreds that have been published, provide a starting point for further reading on various aspects of reactive transport modeling and its applications. In addition to these citations, you may be interested in the references listed in the *GWB Essentials Guide* and the *GWB Reaction Modeling Guide*.

C.1 Reactive transport

Bahr, J.M. and J. Rubin, 1987, Direct comparison of kinetic and local equilibrium formulations for solute transport affected by surface reactions. *Water Resources Research* **23**, 438–452.

Bethke, C.M., 1997, Modelling transport in reacting geochemical systems. *Comptes Rendus de l'Académie des Sciences* **324**, 513–528.

Bethke, C.M., 2022, *Geochemical and Biogeochemical Reaction Modeling*, 3rd ed. Cambridge University Press, New York, 520 p.

Bethke, C.M. and P.V. Brady, 2000, How the K_d approach undermines groundwater cleanup. *Ground Water* **38**, 435–443.

Knapp, R.B., 1989, Spatial and temporal scales of local equilibrium in dynamic fluid-rock systems. *Geochimica et Cosmochimica Acta* **53**, 1955–1964.

Lichtner, P.C., 1985, Continuum model for simultaneous chemical reactions and mass transport in hydrothermal systems. *Geochimica et Cosmochimica Acta* **49**, 779–800.

Lichtner, P.C., C.I. Steefel and E.H. Oelkers (eds.), 1996, Reactive transport in porous media. *Reviews in Mineralogy* **34**, 438 p.

Shevaliera, M., C. Dalkhaa, P. Humez, B. Mayer, V. Becker, M. Nightingale, Luc Rock, and G. Zhang, 2014, Coupling of TOUGHREACT-Geochemist Workbench (GWB) for modeling changes in the isotopic composition of CO₂ leaking from a CCS storage reservoir. *Energy Procedia* **63**, 3751–3760.

Steefel, C.I., D.J. DePaolo and P.C. Lichtner, 2005, Reactive transport modeling: An essential tool and a new research approach for the Earth sciences. *Earth and Planetary Science Letters* **240**, 539–558.

Steefel, C.I. and A.C. Lasaga, 1994, A coupled model for transport of multiple chemical species and kinetic precipitation/dissolution reactions with application to reactive flow in single phase hydrothermal systems. *American Journal of Science* **294**, 529–592.

Stumm, W., 1992, *Chemistry of the Solid-water Interface*. Wiley, New York, 428 p.

Valocchi, A.J., 1985, Validity of the local equilibrium assumption for modeling sorbing transport through homogeneous soils. *Water Resources Research* **21**, 808–820.

Valocchi, A.J., R.L. Street, and P.V. Roberts, 1981, Transport of ion-exchanging solutes in groundwater, chromatographic theory and field simulation. *Water Resources Research* **17**, 1517–1527.

Yeh, G.T. and V.S. Tripathi, 1989, A critical evaluation of recent developments in hydrogeochemical transport models of reactive multi-chemical components. *Water Resources Research* **25**, 93–108.

C.2 Fluid viscosity

Phillips, S.L., H. Ozbek, A. Igbene and G. Litton, 1980, Viscosity of NaCl and other solutions up to 350°C and 50 MPa pressures. *Lawrence Berkeley Laboratory Report LBL-11586*, 71 p.

C.3 Silicate kinetics and weathering

Blum, A.E. and L.L. Stollings, 1995, Feldspar dissolution kinetics. *Reviews in Mineralogy* **31**, 291–351.

Carroll, S.A. and J.V. Walther, 1990, Kaolinite dissolution at 25°C, 60°C, and 80°C. *American Journal of Science* **290**, 797–810.

Leamson, R.N., J. Thomas, Jr. and H.P. Ehrlinger, III, 1969, A study of the surface areas of particulate microcrystalline silica and silica sand. *Illinois State Geological Survey Circular* **444**, 12 p.

Nagy, K.L., 1995, Dissolution and precipitation kinetics of sheet silicates. *Reviews in Mineralogy* **31**, 173–233.

Nagy, K.L. and A.C. Lasaga, 1992, Dissolution and precipitation kinetics of gibbsite at 80°C and pH 3, the dependence on solution saturation state. *Geochimica et Cosmochimica Acta* **56**, 3093–3111.

Rimstidt, J.D. and H.L. Barnes, 1980, The kinetics of silica-water reactions. *Geochimica et Cosmochimica Acta* **44**, 1683–1700.

White, A.F., 1995, Chemical weathering rates of silicate minerals in soils. *Reviews in Mineralogy* **31**, 406–461.

C.4 Steam flooding

Hutcheon, I., 1984, A review of artificial diagenesis during thermally enhanced recovery. In D.A. MacDonald and R.C. Surdam (eds.), *Clastic Diagenesis*. American Association of Petroleum Geologists, Tulsa, 413–429.

C.5 MODFLOW-2000

Harbaugh, A.W., E.R. Banta, M.C. Hill and M.G. McDonald, 2000, MODFLOW-2000, the U.S. Geological Survey modular groundwater model—User guide to modularization concepts and the ground-water flow process. *U.S. Geological Survey Open-File Report* 00-92, 121 p.