

# *mfLab* how-to's

Theo Olsthoorn

5 April 2010

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Making a grid</b>	<b>2</b>
<b>3</b>	<b>MULTIDIFFUSION</b>	<b>3</b>
<b>4</b>	<b>Modeling heat transport</b>	<b>3</b>
4.1	With flow . . . . .	4
4.1.1	Example . . . . .	5
4.2	Die-out after an initial temperature profile . . . . .	6
4.3	Sudden load of mass . . . . .	7
4.3.1	Injection pulse with mass loading . . . . .	10
4.4	Reheating of a geothermal system . . . . .	11
<b>5</b>	<b>Boundary conditions for transport</b>	<b>17</b>
5.1	Constant concentration cells cannot be switched off, alas!! . . .	19
<b>6</b>	<b>Understanding Seawat input for viscosity and density</b>	<b>19</b>
6.1	Boundary conditions for constant head with variable density . . .	19
<b>7</b>	<b>Steady-state versus transient flow with transport</b>	<b>20</b>
7.1	Viscosity in the NAM file with density package off . . . . .	21
7.2	Density package . . . . .	21
7.2.1	MT3DRHOFLAG ( $\rho Flag$ ) . . . . .	21
<b>8</b>	<b>Viscosity package</b>	<b>22</b>
8.0.2	MT3DMUFLAG ( $\mu Flag$ ) . . . . .	22
8.0.3	MUTEMPOPT ( $\mu$ temperature option) . . . . .	26

## 1 Introduction

*mfLab* is a strong modeling concept and environment by combining the strengths of Matlab and the standard and robust finite difference groundwater flow and

transport models MODFLOW, MT3DMS SEAWAT and the wealth of packages made for them. However, to become a skilled modeler, most of us need examples and be shown how to do things in this environment efficiently. This may prevent a lot of frustration.

mfLab models are typically made by copying an existing example and adapting it to one's needs. Therefore, we need examples of a wide spectrum of usage. Many examples can be found in the examples directory that comes with *mfLab*. These examples are generally well documented with interspersed comments. However, all too many comments also distracts from the essence and makes m-files needlessly long. Therefore, some basic skills in mfLab modeling or modeling in Matlab in general, can best be taught from a separate how-to manual, that this one tries to present.

The subjects have not been made nor ordered in a systematic way. Rather they have been made while modeling myself. This manual will therefore, be extended regularly when new approach are made and examples generated.

## 2 Making a grid

A grid can be made by hand, by typing in numbers, or by using available functions, or by a combination. *mfLab* requires an **xGr**, **yGr** and a **zGr** vector. **zGr** may also be a complete 3D array. In that case, **zGr** specifies the top and bottom of each and every cell in the model.

Having specified the grids, there are two functions to make sure that **xGr**, **yGr** and **zGr** are sorted, oriented in the right vector direction and that duplicates are removed.

```
[xGr,yGr,xm,ym,Dx,Dy,Nx,Ny]=modelsizex(xGr,yGr);
```

or

```
[xGr,yGr,zGr,xm,ym,zm,Dx,Dy,Dz,Nx,Ny,Nz]=modelsizex3(xGr,yGr,zGr);
```

the **m** in **xm**, **ym**, and **zm** indicates the vectors of the cell centers, the **D** in **Dx**, **Dy** and **Dz** indicate the vectors of cell sizes and the **N** in **Nx**, **Ny** and **Nz** indicate the number of cells in the corresponding grid directions.

**yGr** and **ym** vectors will be oriented high to low, so that the first line in the arrays correspond with the most northerly position. Likewise, **zGr** and **zm** are oriented from high to low to make sure that the first line (or first plane) of an array is the layer with the highest elevation.

To facilitate making a grid, rather than typing numbers you may use Matlab's **linspace** and **logspace** functions. See their documentation.

*mfLab* comes with the function **sinspace** to add details and generate smooth transition between parts of the grid around objects (see directory mflab/mfiles/gridcoords) and type **help sinspace**.

```
[x,dx]=sinspace(x1,x2,N,alfa1 [,alfa2])
```

divides the axis between and including **x1** and **x2** into **N+1** sections, **N** gridlines, with section lengths according to the the sine function. If **alfa2** is left, out it is interpreted as **alfa1=0** and **alfa2=alfa1**. So to refine the grid towards **x2**

```

[x,dx]=sinspace(x1,x2,N,pi/2,pi); % refines the grid towards x2
[x,dx]=sinspace(x1,x2,N,0,pi/2); % refines the grid towards x1
[x,dx]=sinspace(x1,x2,N,0,pi); % refines the grid towards x1 and x2,
coarse in the middle

```

If grid coordinates are generated in arbitrary ways, involving many a fine grid around wells for instance inside a coarser grid, that itself also honours the details of a local stream end so on, then, one all these coordinates are put together, one may expect a very irregular grid, not only with duplicates but, especially also with near duplicates. Such small cells are rather merged with larger neighboring cells to make sure no cells are smaller than some specified minimum cell size. The function `cleangrid` can do the job (be it in a bit simplistic way). Especially the computation time of transport models strongly depends on the minimum cell size.

```

xGr=cleangrid(xGr,dxmin);

```

This can be repeated for the yGr and zGr directions in the same way, using `dyzim` and `dzmin`.

### 3 MULTIDIFFUSION

Diffusion coefficients are specified in the LAY sheet of the workbook under the heading DMCOEF, hence per layer. However, MT3DMS allows DMCOEF to be specified on a cell-by-cell basis. Therefore, you can specify DMCOEF as a parameter in *mfLab* workspace as well as in the LAY worksheet. The parameter DMCOEF must be a cell array of length NLAY that has a matrix NROW,NCOL of diffusion coefficient values for each layer that needs to be specified. Which layers to specify in the workspace is deduced from the column DMCOEF in the LAY sheet. If the value of a layer in the worksheet is  $\geq 0$ , then the value in the worksheet is used. If, however this value is  $< 0$ , then the value in the cell array is used, taking the cell that corresponds to the layer being processed. If DMCOEF in a layer in the worksheet  $< 0$ , *mfLab* requires the matrix DMCOEF to be present in the workspace. Clearly, layers that are specified in the worksheet LAY may correspond to empty cells in the cell array DMCOEF.

For convenience of the user, *mfLab* also accepts a regular array of diffusion coefficients of size (NROW, NCOL, NLAY, NCOMP) where NCOMP the number of species each with its own diffusion coefficient.

### 4 Modeling heat transport

Heat is treated mathematically the same as diffusion combined with sorption. Hence, we need to specify both the diffusion coefficient of the cells on a per layer or per cell basis as well as the sorption process. The diffusion coefficient is specified in the LAY worksheet on a per layer basis, which may be mixed with coefficients on a per cell basis, which is given as the parameter DMCOEF in *mf\_adapt*. *mfLab* uses the *mf\_adapt* values if DMCOEF in the LAY-sheet

of a given layer has a negative value. See section 3 on the preceding page. Diffusion coefficients are handed over to MT3DMS and SEAWAT through the DSP (dispersion-diffusion) package and sorption coefficients through the RCT (reaction) package.

To model conduction-convection of heat with MT3DMS or SEAWAT, we must compare the mathematical formulations. If we are merely interested in pure diffusion and pure conduction, so no flow (no dispersion and no convection), there is no need for the reaction package. This can be shown as follows. The mass balance equation and heat balance equation are then, noting that in the left equation  $c$  is concentration and in the right equation  $c$  is heat capacity (!) :

$$\epsilon \left( 1 + \frac{\rho_b K_d}{\epsilon} \right) \frac{\partial c}{\partial t} = \epsilon D_s \nabla^2 c, \quad \rho c \frac{\partial T}{\partial t} = \lambda \nabla^2 T$$

so that

$$\left( 1 + \frac{\rho_b K_d}{\epsilon} \right) \frac{\partial c}{\partial t} = D_s \nabla^2 c, \quad \rho c \frac{\partial T}{\partial t} = \lambda \nabla^2 T$$

Hence, both systems are equivalent if we set

$$D_{\text{diff}} = \frac{D_s}{1 + \frac{\rho_b K_d}{\epsilon}} = \frac{\lambda}{\rho c}$$

and simulate only diffusion, without sorption, or, equivalently, set  $D_{\text{diff}} = D_s = \lambda/\rho c$ , with  $1 + \rho_b K_d/\epsilon = 1$ , so that  $K_d = 0$ . That is, use linear sorption but set the distribution coefficient equal to zero.

Alternatively, we may set

$$D_{\text{diff}} = D_s = \lambda, \quad K_d = \epsilon \frac{(\rho c - 1)}{\rho_b} \simeq \epsilon \frac{\rho c}{\rho_b}$$

That is use the bulk heat conductance as diffusion coefficient and  $\epsilon(\rho c)/\rho_b$  as distribution coefficient.

#### 4.1 With flow

In the case we have flow so that advection (and, therefore, dispersion) and or convection are working, we have to include those processes in the heat and mass balance:

$$\epsilon \left( 1 + \frac{\rho_b K_d}{\epsilon} \right) \frac{\partial c}{\partial t} = \epsilon D \nabla^2 c - \epsilon v \nabla c, \quad \rho c \frac{\partial T}{\partial t} = \lambda \nabla^2 T - \epsilon \rho_w c_w v \nabla T$$

so that the left equation leads to

$$\frac{\partial c}{\partial t} = \frac{D}{R} \nabla^2 c - \frac{v}{R} \nabla c$$

Now we make the right-hand equation equivalent to the left hand one

$$\rho c \frac{\partial T}{\partial t} = \lambda \nabla^2 T - \epsilon \rho_w c_w v \nabla T$$

using  $\rho c = \epsilon \rho_w c_w + \rho_b c_s$ , and dividing left and right by  $\rho_w c_w$ , we get

$$\epsilon \left( 1 + \frac{\rho_b c_s}{\epsilon \rho_w c_w} \right) \frac{\partial T}{\partial t} = \frac{\epsilon \lambda}{\epsilon \rho_w c_w} \nabla^2 T - \epsilon \frac{\rho_w c_w}{\rho_w c_w} v \nabla T$$

$$\epsilon \left( 1 + \frac{\rho_b K_d}{\epsilon} \right) \frac{\partial T}{\partial t} = \epsilon D_H \nabla^2 T - \epsilon v \nabla T, \quad K_d = \frac{c_s}{\rho_w c_w}, \quad D_H = \frac{\lambda}{\epsilon \rho_w c_w}$$

so that

$$\frac{\partial T}{\partial t} = \frac{D_H}{R} \nabla^2 T - \frac{v}{R} \nabla T$$

So that equivalence is achieved when setting

$$D_H = \frac{\lambda}{\epsilon \rho_w c_w}, \text{ and } K_d = \frac{c_s}{\rho_w c_w} \text{ and } R = 1 + \frac{\rho_b K_d}{\epsilon} = 1 + \frac{\rho_b c_s}{\epsilon \rho_w c_w} = \frac{\rho c}{\epsilon \rho_w c_w}$$

As can be seen upon inspection of the right-hand expression, the retardation in the case of heat transport is the total heat capacity of a m<sup>3</sup> of porous medium including water over the heat capacity of the water in the pores, i.e. the present water. This is always the case: the retardation is the total mass per m<sup>3</sup> of porous medium over the mass dissolved in the present water, the porosity times the concentration in the water. The equivalent distribution coefficient in the case of heat is now also known. It adheres exactly to the definition of the distribution coefficient. Namely, given water a certain temperature, the the heat stored in a m<sup>3</sup> of this water is  $\rho_w c_w$ , while the heat stored in a kg of solids with the same temperature is  $c_s$ .

Also the equivalent diffusivity  $D = \frac{\lambda}{\epsilon \rho_w c_w}$  can be physically understood. In diffusion/dispersion  $D$  is the total mass flux through the pores driven by the concentration gradient. In the case of heat  $\lambda$  is the total heat flux through both pores and solids. To make this heat flux comparable with the diffusion/dispersion case, then we do as if this heat flux is through the pores, which translates into a temperature flux equal to  $\lambda/(\epsilon \rho_w c_w)$ .

In simulating heat, the latter case is the most general, as it allows for ground-water flow to be combined with heat flow. To simulate heat with MT3DMS or SEAWAT we thus have to specify

$$D_{\text{diff}} = \frac{\lambda}{\epsilon \rho_w c_w}, \quad K_d = \frac{c_s}{\rho_w c_w}$$

#### 4.1.1 Example

For ordinary value of the parameters we may set

$$\lambda = \epsilon \lambda_w + (1 - \epsilon) \lambda_s$$

assuming  $\epsilon = 0.35$ ,  $\lambda_w = 0.6 \text{ W/m/K}$  and  $\lambda_s = 3 \text{ W/m/K}$ , we have

$$\lambda = 0.35 \times 0.6 + (1 - 0.35) 3.0 = 2.16 \text{ W/m/K} = 0.19 \times 10^6 \text{ J/d/m/K} = 68 \times 10^6 \text{ J/y/m/K}$$

Bulk heat capacity is

$$\rho c = \epsilon \rho_w c_w + (1 - \epsilon) \rho_s c_s$$

with  $\rho_w = 1000 \text{ kg/m}^3$ ,  $\rho_s = 2650 \text{ kg/m}^3$ ,  $c_w = 4200 \text{ J/kg/K}$ ,  $c_s = 800 \text{ J/kg/K}$ ,

$$\rho c = 0.35 \times 1000 \times 4200 + (1 - 0.35) \times 2650 \times 800 = 2.85 \times 10^6 \text{ J/m}^3/\text{K}$$

$$D_{\text{diff}} = D_H = \frac{\lambda}{\epsilon \rho_w c_w} \simeq \frac{0.19 \times 10^6}{0.35 \times 4.2 \times 10^6} = 0.13 \text{ m}^2/\text{d} = 47 \text{ m}^2/\text{d}$$

$$K_d = \frac{c_s}{\rho_w c_w} = \frac{800}{4.2 \times 10^6} = 1.9 \times 10^{-4} = \frac{1}{5250} \frac{\text{J/kg}_{\text{solids}}}{\text{J/m}^3_{\text{water}}}$$

The dimension being the heat per kg solids versus the heat per  $\text{m}^3$  water of the same temperature.

For the retardation of the heat front we have:

$$R = 1 + \frac{\rho_b K_d}{\epsilon} = 1 + \frac{(1 - 0.35) \times 2650 \times 1.9 \times 10^{-4}}{0.35} = 1.94$$

$$R = \frac{\rho c}{\epsilon \rho_w c_w} = \frac{2.85 \times 10^6}{0.35 \times 4.2 \times 10^6} = 1.94$$

So that heat fronts travel with approximately half the velocity of water.

In the model we have to set  $D_H = 0.13 \text{ m}^2/\text{d}$  in the LAY worksheet and  $SP1 = K_d = 1.9 \times 10^{-4} \text{ m}^3/\text{kg}$  in the LAY spreadsheet under column SP1. In the MT3D worksheet we need to specify linear adsorption through the parameter ISOTHM. For linear sorption  $ISOTHM = 0$  and  $SP1 = K_d$  while  $SP2$  is read but not used.

Figure 1 shows the results of heat conduction from a constant temperature source at  $x = 0$ . The parameters used are as given above. The model was run in steady state, there is no groundwater flow, so pure conduction. The markers are the analytical solution results as

$$c = c_0 \text{erfc}\left(\frac{x}{\sigma \sqrt{2}}\right)$$

which, for  $\sigma = z$  always yields  $c = c_0 \text{erfc}(1/\sqrt{2}) = 0.32$ , while  $\sigma = x = \sqrt{2 \frac{D_H}{R} t}$  was used to place these markers to verify the numerical results. The model was a single row model with 2000 cells in  $x$ -direction.

## 4.2 Die-out after an initial temperature profile

This situation can be simulated in at least two ways, one is to setup an initial concentration which will die out as time passes. To this end we used a start

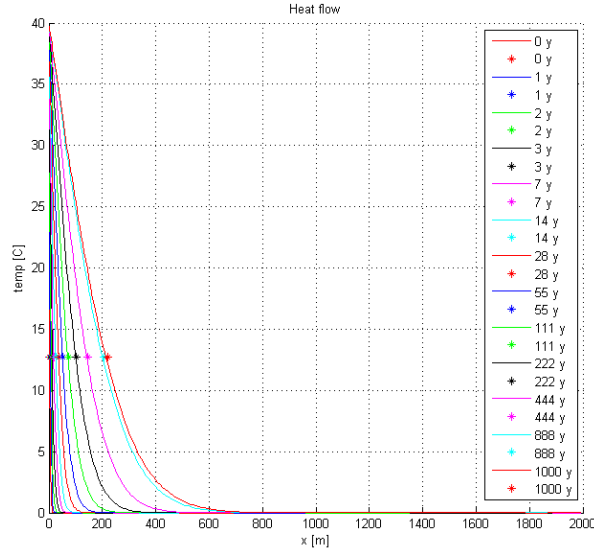


Figure 1: Heat conduction from a fixed temperature

temperature equal to zero, except for distances smaller than  $W/2$  (half aquifer width), where the start concentration was set equal to the initial temperature of 40C.

The analytical solution is

$$T = \frac{T_0}{2} \left( \operatorname{erfc} \left( \frac{x - W/2}{\sigma \sqrt{2}} \right) - \operatorname{erfc} \left( \frac{x + W/2}{\sigma \sqrt{2}} \right) \right), \quad \sigma = \sqrt{2 \frac{D_H}{R} t}$$

The results are given in figure 2. The drawn lines are the results of the numerical computation with MT3DMS and the dots are the analytically computed values. There is a small deviation between the two for  $x < W/2$  but which gradually disappears with time. I don't know the reason of this small deviation. It is not the grid accuracy because five times more grid points were used than points for the analytical computation.

### 4.3 Sudden load of mass

Although this situation is clear, it is more tricky to solve numerically. Here we have to use two stress periods, one to inject the mass and a much longer one to let the concentration profile die out. Here the first stress period is 1 day and the second is 10 years, 36500 days.

The analytical solution for a initial mass  $M$  which dies off afterwards is given by

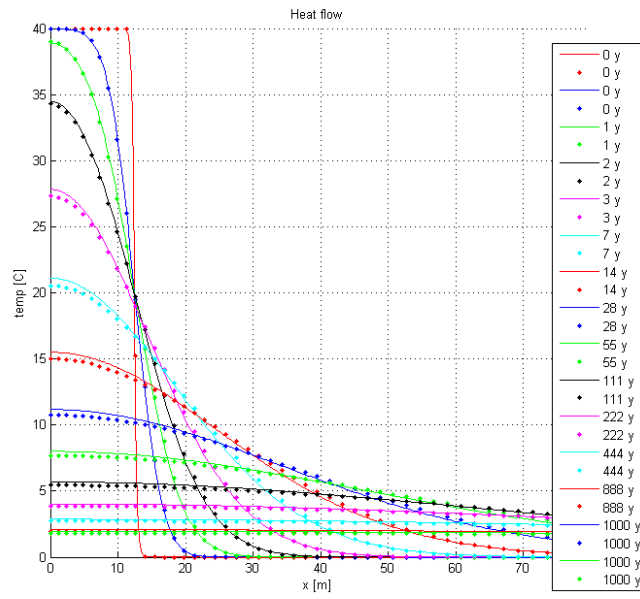


Figure 2: Die-out of an initial temperature of 40C (data used are given in the text)



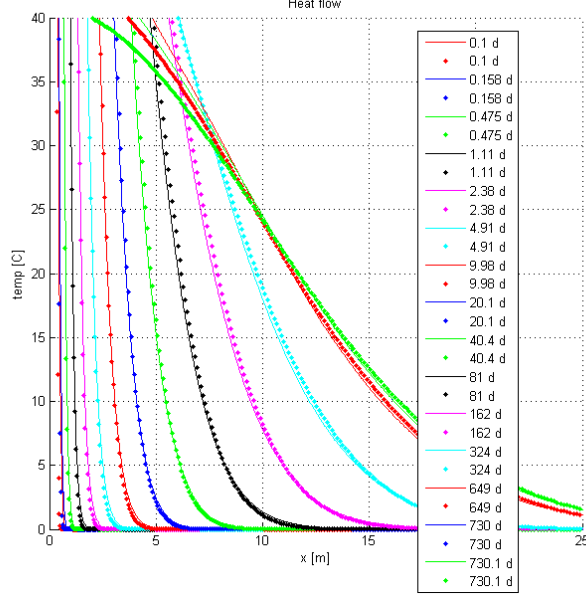


Figure 3: Result of heat pulse with the same total heat as in the case of an initial temperature over the width of the aquifer shown in figure 2

$$c = \frac{M}{\epsilon R} \frac{e^{-\frac{x^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}$$

To compare this mass with a concentration and an application width se set  $M = \epsilon R c_0 W$ . Transferring the concentration to temperatures yields this:

$$T = \frac{\epsilon R T_0 W}{\epsilon R} \frac{e^{-\frac{z^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}} = \frac{T_0 W}{\sigma\sqrt{2\pi}} e^{-\frac{z^2}{2\sigma^2}} \quad (1)$$

To compare with the previous case, we set  $T^* = T_0(W/2)/dx = 40 \times 12.5/0.225 = 427 \text{ C}$  as initial temperature in the first cell of the model, which has a width 0.225 m. This will store the same initial amount of heat in this cell as previously stored in the half width of the aquifer at  $T_0 = 40 \text{ K}$ . But the analytical solution remains as in 2, no model involved.

The results are given in the two figures below, one for short times after the release of heat and one for long times. The maximum temperature scale is set to 40C for easy comparison with the previous result.

Both figures show a good agreement with the analytical solution.

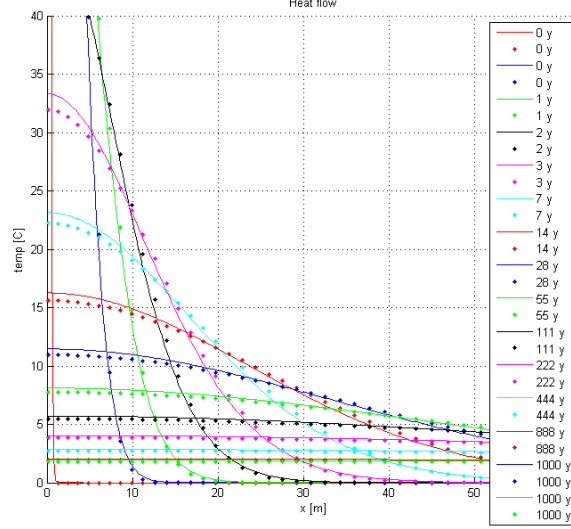


Figure 4: Result of heat pulse with same total heat as in the case of an initial temperature of 40C over the width of the 25 m thick aquifer in figure 2

#### 4.3.1 Injection pulse with mass loading

The last example is a pulse injection at time zero with the same amount of heat as was the initial situation in the previous cases. But this time the injection is provide by means of a mass loading, a direct injection of heat. This option can be used by setting ITYPE=15 in the point sources specified for the SSM package of MT3DMS.

$$T = \frac{M}{\epsilon R \sigma \sqrt{2\pi}} e^{-\frac{z^2}{2\sigma^2}} \quad (2)$$

The model solves equation 2 and thus provides for division by  $\epsilon R$ . The M in terms of temperature equals

$$M = \epsilon R T_0 W$$

and the mass loading such that during the first stress period this mass is injected in the first cell of the model is

$$\frac{dM}{dt} = \frac{\epsilon R T_o W}{dt}$$

with  $dt$  the length of the first stress period.

This situation is case 5 of the example. The results are shown in the figure

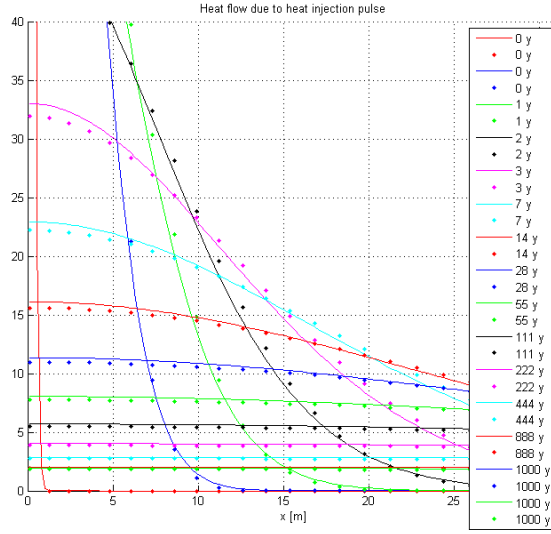


Figure 5: Temperatures after a sudden injection of heat at  $t=0$  in the first cell of the model using mass loading (ITYPE=15 in the SSM package).

#### 4.4 Reheating of a geothermal system

Geothermal systems are claimed to be a sustainable future promise. Such systems extract hot groundwater from an appropriate depth, use the heat and inject the cooled water back into the same aquifer (layer) at a suitable distance, such that the cooled water does not reach the hot extraction well during the projected lifespan of the system. The heat is the temperature due to the normal geothermal gradient of about 30K/km. Hence at a depth of around 2 km, temperature in the order of 70C may be expected.

In favorable circumstances, these the temperature may be higher. The distance between the hot and cold well may be in the order of 2000 m, the thickness of the layer, often a sandstone about 20% and its thickness in the order of 100 m. A suitable layer may bend up- and downward under past tectonic movements in the earth's crust (figure 6). In such cases it is favorable to extract from the deeper, hotter, elevation and re-inject into the higher elevation to save drilling cost. The flow between the two wells is subject to heterogeneities and possible faults in the crust and layer. But this flow is also subject to viscosity effects, as the cooled water has a much higher viscosity than the original hot water, and it will be subject to density effects as the cooled water has a higher density than the hot water.

These effects make the flow complicated and careful study of the properties of the subsurface layers and flow processes are necessary for a good and safe design of a geothermal systems. We pass over all such detail here and ask

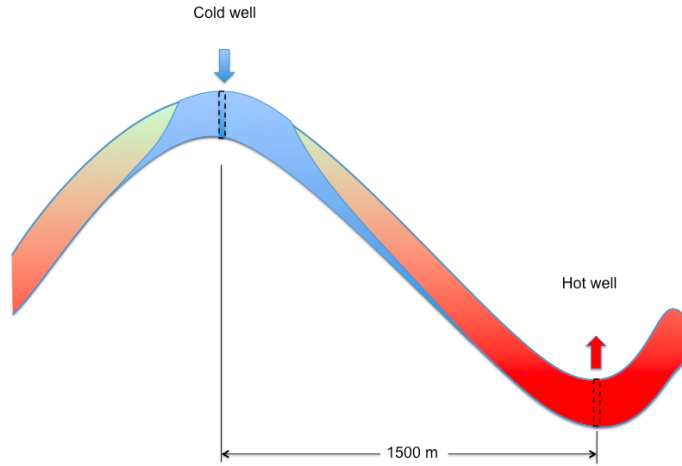


Figure 6: Impression of a geothermal aquifer (x-section) with extraction and injection well and spreading of cooled water subject to density and viscosity effects

ourselves how sustainable geothermal systems are. That is, how long does it take for the layers from which the heat was extracted until they are reheated again naturally and can be reused. Is this 1, 10, 100, 1000 or 10000 years?

The cold front spreads out from the cold well to finally reach the hot well. After the cold front has passed a point in the aquifer, the adjacent over- and underlying layers will be cooled by the “cold” water in the geothermal aquifer (see figure 7). The duration of this cooling depends on the time since the passing of the cold front. Therefore, at the end of the lifetime of the geothermal system, adjacent layers near the injection well have been subject to cooling during the entire lifetime of the system, while near the front this cooling time is zero. The figure given an impression of this situation. It shows the temperatures in the geothermal aquifer between the injection cold well and the extraction hot well and also the cooling of the adjacent layers above and below. Hence, at the end of the lifetime or life-cycle of a geothermal system, for any point there is a certain time since the cold front passed and cooling of the adjacent layers has been proceeding, additionally the temperature in the geothermal aquifer is equal to the injection temperature. We will answer the question how long reheating takes for such a point.

Reheating is, in fact not a good concept. What happens is that the temperature anomaly due to the injection of relatively cold water is superimposed upon the natural initial temperature, at least during the time that the boundary temperature at the surface of the earth plays no role. That is the time during which the temperature anomaly does not reach ground surface. As this time is very long, it may be neglected at first, only to be checked later. If the effect of the temperature boundary at ground surface is negligible on the temperature

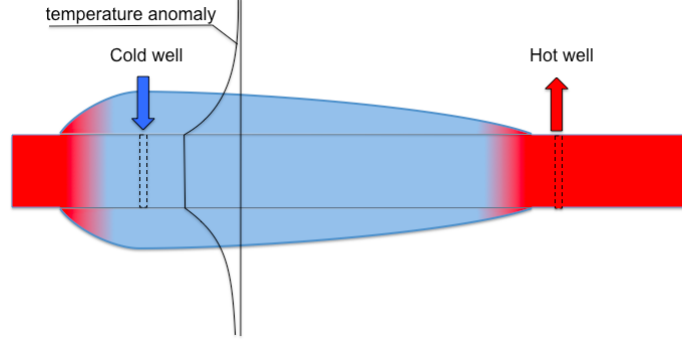


Figure 7: Geothermal aquifer (x-section) with cooled injection fluid moving towards the extraction well, while also cooling the overlying and underlying layers. Effects of density and viscosity ignored.

distribution around our geothermal system, then reheating is not the right word. What takes place is a redistribution of the anomalous temperature under heat conduction, if, as we do here, influence of groundwater flow in these over- and underlying layers can be neglected. If they are not highly permeable, this is generally true.

Hence we consider the loss of energy to (gain of heat from) the layers above the geothermal aquifer, considering its top as a constant temperature layer during the time between the passage of the cold front and the end of the life of the system. The same is valid for the layers below the geothermal aquifer. Then, given the heat distribution thus obtained at the end of the system's life, we compute the dissipation over time, also taking into consideration the temperature anomaly in the aquifer itself, which dissipates from the point onwards.

Because of the principle of superposition we deal with only the temperature change  $T_0 = 50C$  of the injection water compared to the original groundwater in the geothermal layer. The loss of heat into overlying or underlying layers from a constant temperature source follows from

$$T = T_0 \operatorname{erfc} \left( \frac{z}{\sigma \sqrt{2}} \right), \quad \sigma = \sqrt{2 \frac{D_H}{R} t}$$

in which

$$D_H = \frac{\lambda}{\epsilon \rho_w c_w}, \quad R = 1 + \frac{\rho_b c_s}{\epsilon \rho_w c_w} = \frac{\rho c}{\epsilon \rho_w c_w}$$

$$\lambda = \epsilon \lambda_w + (1 + \epsilon) \lambda_s$$

$$\rho c = \epsilon \rho_w c_w + \rho_b c_s = \epsilon \rho_w c_w + (1 + \epsilon) \rho_s c_s$$

The heat loss can be compute from this analytical solution as follows:

$$q_H = -\lambda \frac{\partial T}{\partial x} = \frac{\lambda}{\sigma} \sqrt{\frac{2}{\pi}} e^{-\frac{z^2}{2\sigma^2}}$$

At  $z=0$ , where the temperature is fixed, the total heat lost (or gained) is

$$H = \lambda T_0 \sqrt{\frac{2}{\pi}} \int_0^t \frac{1}{\sqrt{2 \frac{D_H}{R}}} t^{-1/2} dt$$

$$H = \lambda T_0 \sqrt{\frac{2}{\pi}} \frac{2\sqrt{t}}{\sqrt{2 \frac{D_H}{R}}} = T_0 \sqrt{\frac{2}{\pi}} \frac{2\lambda t}{\sigma} = \epsilon \rho_w c_w R T_0 \sqrt{\frac{2}{\pi}} \frac{2 \frac{D_H}{R} t}{\sigma} = \rho c T_0 \sigma \sqrt{\frac{2}{\pi}}$$

$$\frac{H}{\rho c} = T_0 \sigma \sqrt{\frac{2}{\pi}}$$

where time is encapsulated in  $\sigma$ . Further,  $H/\rho c$  is an equivalent thickness containing the same energy at  $T = T_0$  as does the real system specified by  $\sigma$ .  $H$  is the total heat lost into the adjacent layers and, therefore, also the amount of (anomalous) heat present in these layers between the boundary and infinity.

The total amount of anomalous heat at this point of the geothermal aquifer between  $\pm\infty$  equals

$$H_T = 2H + \rho c T_0 W = \rho c T_0 \left( 2\sigma \sqrt{\frac{2}{\pi}} + W \right) \quad (3)$$

If  $t$  in  $\sigma$  equals the time between the passing of the cold front and the end of the life of the geothermal system, then  $H_T$  is the total amount of anomalous heat stored as this point in the aquifer between  $-\infty \leq z \leq \infty$ , which will dissipated after the system has been abandoned. Thus,  $\sigma$  in equation 3 is a fixed value after the system stopped, we write further  $\sigma_0$  for it.

Dissipation of heat from a sudden source is given by the following analytical solution

$$T = \frac{M}{\epsilon R \sigma \sqrt{2\pi}} e^{-\frac{z^2}{2\sigma^2}}, \quad M = \epsilon R T^* dx$$

Where  $T^* dx$  the given initial temperature and  $dx$  the width over with this temperature is specified. A true pulse is where  $dx \rightarrow 0$  and  $T^* dx = \text{constant}$ . For very long times, the initial distribution of the heat around  $z = 0$  is of little importance, the distribution will gradually approach a the bell shape of the Gaussian normal probability density function. Therefore, for long times, we may ignore this initial distribution and consider the entire amount of anomalous heat in equation 3 as a single pulse at  $t = 0$ .

Hence

$$T = \frac{T_0 \left( \sigma_0 \sqrt{\frac{8}{\pi}} + W \right)}{\sigma \sqrt{2\pi}} e^{-\frac{z^2}{2\sigma^2}},$$

And the temperature in the center of the aquifer thus becomes

$$\frac{T}{T_0} = \frac{\sigma_0 \sqrt{\frac{8}{\pi}} + W}{\sigma \sqrt{2\pi}}$$

The reheating time of the geothermal system may be equated to the time it takes until  $T/T_0 = 0.05$ , so that 95% of the heat anomaly has disappeared by dissipation of the heat anomaly into the overlying and underlying layers:

$$\sigma = \left( \frac{T_0}{T} \right) \left( \frac{\sigma_0 \sqrt{\frac{8}{\pi}} + W}{\sqrt{2\pi}} \right)$$

from which

$$t = \frac{\sigma^2}{2} \frac{R}{D_H}$$

We may model this process in mflab using a column of cells from ground surface to somewhere deep below the geothermal aquifer. We may then compute the initial situation analytically exact, or as pulse containing all anomalous heat lost at this point since the cold front passed by. Either method is accurate after long times, say 5 to 10 times the life span of the system. For visualisation purposes it may be nice to start with the analytical solution at the moment that the system is stopped and superpose this on the natural geothermal gradient. This will be done in this example.

The natural temperature gradient starts at say  $T_0 = 10^\circ\text{C}$  and increases by  $G = -30 \text{ K/km}$ . Taking  $z$  upward positive and the center of the aquifer at  $Z_0$ ,

$$T = T_0 - Gz$$

Between the top and bottom of the aquifer  $Z_0 - W/2 \leq Z \leq Z_0 + W/2$  we have  $T = T_{Z_0} - \Delta T$  and above and below we superimpose

$$\begin{aligned} \Delta T &= \Delta T_{\text{Top}} \text{erfc} \left( \frac{z - (Z_0 + W/2)}{\sigma_0 \sqrt{2}} \right), \quad z \geq Z_0 + W/2 \\ \Delta T &= \Delta T_{\text{bot}} \text{erfc} \left( \frac{-z + (Z_0 - W/2)}{\sigma_0 \sqrt{2}} \right), \quad z \leq Z_0 - W/2 \end{aligned}$$

With this initial temperature distribution we may compute the development over time using MT3DMS (or SEAWAT) with a single column of cells of  $1 \text{ m}^2$  cross section. For convenience of plotting the  $y$  direction was chosen instead of  $z$  for this column. The column has 4000 cells in  $y$  direction. It is not feasible to make a model with 4000 layers instead. The input will then be much more extended and I'm not sure whether such a model will actually work. But a model consisting of a single column of 4000 cells in  $y$  direction was no problem at all for MODFLOW or MT3DMS.

The temperature at the top and bottom of the model have been fixed during this simulation. This is OK for the top but perhaps less so for the bottom.

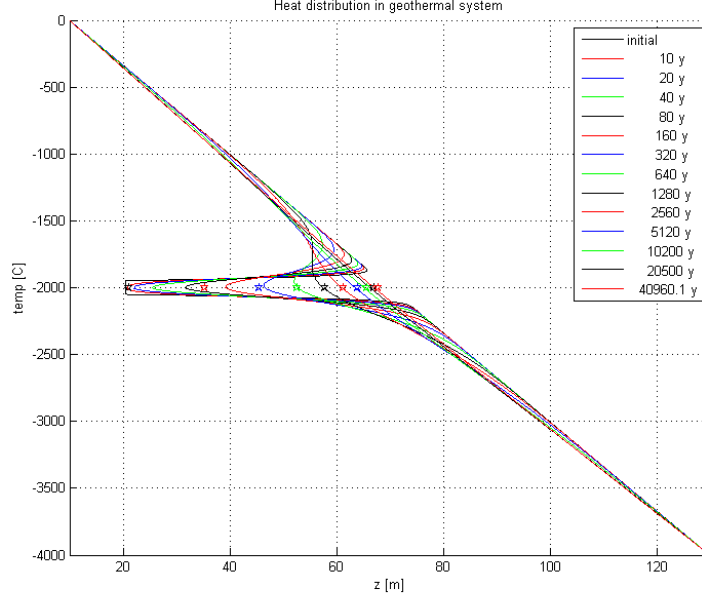


Figure 8: Temperature distribution and development after usage of a layer for geothermal heat extraction. The stars are the analytical solution temperatures at the center of the geothermal aquifer.

Nevertheless, the bottom is so far away from the geothermal aquifer that it will have no influence on the conclusions.

Note that this model has no groundwater flow, only heat conduction is taken into account.

The results are shown in figure 8, which demonstrates that reheating or rather the redistribution of the temperature anomaly caused by the use of the heat of a geothermal aquifer will take several tens of thousands of years in this case. The results of the analytical solution, i.e. the temperature at the center of the geothermal aquifer are also shown. Clearly, during the first years this solution does not match the numerical one because the initial temperatures differ a lot. But after about 300 years the two match accurately (figure 9). At the end the difference increases a bit due to the influence of the boundary conditions at the top and the bottom of the system, i.e. at 0 and 4000 m depth.

The time of reheating will be shorter if the layer is less thick, for instance several thousand years for a layer of 25 m thick instead of 100 m. With respect to the other parameters, i.e. heat capacity of solids and water nor heat conductance of solids and water, there will not be very much variation, at least no so much that this impression of the reheating time will be invalidated, except, perhaps, convective flows. But even these will take thousands of years.



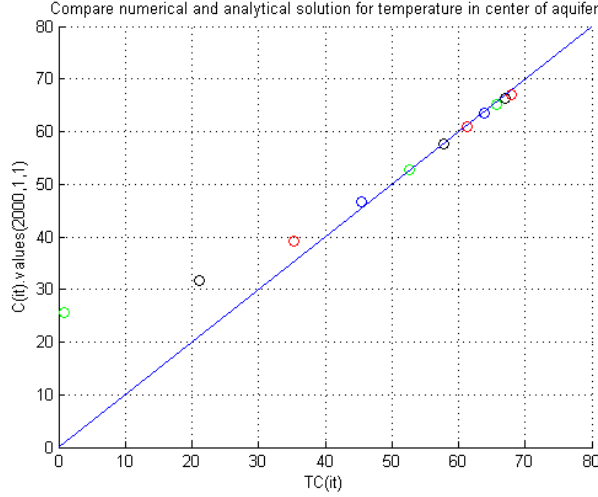


Figure 9: Comparison of the analytical and numerical solution for the temperature at the center of the geothermal aquifer. Small deviations at the end are due to influence of constant temperature boundaries. The points are the same as in the previous figure and span a period from 40-41000 years. The blue circle in the middle is 320 years.

## 5 Boundary conditions for transport

The SSM package of MT3D requires boundary conditions for source-sink terms to be specified unless inflowing water is meant to have concentration zero. Hence boundaries with constant concentration and wells with given concentrations need to be specified. One has the option to use ICBUND to do so, but then these cells will behave as constant concentration cells throughout the simulation. This is a rather rare boundary condition for transport and, therefore not that often used.

For most sources and sinks concentration boundary conditions have thus to be specified. There is an option to do this in the workbook, worksheet PNTSRC (point sources). What is required is a list of

`PER LAYER ROW COL CSS ITYPE CSSMS_1 CSSMS_2 CSSMS_3`

for every cell that is a source or a sink.

where

PER = stress period

LAYER = layer number

ROW = row number

COL = column number

CSS = concentration of species in case only one species is used

ITYPE = type of boundary (fixed concentration, well etc)

ITYPE = 1 constant head cell

ITYPE = 2 is a well

ITYPE = 3 is a modflow drain

ITYPE = 4 is a modflow river

ITYPE = 5 is general head-dependent boundary cell

ITYPE = 15 is a mas loading cell

ITYPE = -1 is a constant concentration cell

CSSMS\_1 .... are concentration of species 1, 2, 3 etc, as far as used. If more than one species is modelled, CSS is dummy but must be present.

Note that *mfLab* requires the stress period number as first item of the list, MT3DMS does not. However, requiring this number facilitates enormously the processing and frees the user of a burden, while it is much more secure. With the stress period number each line is unique and users may mix their input, for instance specifying all stress periods at once for each node instead of all nodes for each stress period at once before proceeding to the following stress period. *mfLab* takes care of sorting if necessary. Further, users are free to leave out data for stress periods. No sequential counting is involved.

When a large number of cells need to be specified, doing so in the spreadsheet is hardly an option. It will be much easier and more flexibly done in *mf\_adapt* inside Matlab. *mfLab* has several functions to facilitate this, mainly ones that translate any part of a 3D cell array into a list as required by the boundary specification.

The function `indices=cellindices(I, dims, orderstr)` converts a list of global index numbers of an array with dimension `dims=size(array)` into a list of cell indices along the dimensions. For instance, we want to specify the PNTSRC required in the BTN package for the top of the model which has constant head. First get the global indices using Matlab's find function

```
Itop=find(Z>zm(1));
```

Where Z is supposed to be the 3D-array with top and bottom of all cells and `zm(1)` the elevation of the center of the topmost cell.

Then using the orderstring LRC to indicate we want layers, columns and rows in that order on each row of the cell index list

```
LRC=cellindices(Itop,size(BOUND),'LRC');
```

Next set the stress period and boundary type numbers and the concentration (temperature) at the boundary

```
iSP=1; iType=1; TempTop=0;
```

Then generate a column of ones of length of I

```
u=ones(size(LRC(:,1)));
```

Then assemble the pointsource list

```
PNTSRC=[u*iSP LCR u*iType u*TempTop];
```

And that's it

The list can be extended with all kinds of other boundaries like

```
PNTSRC=[  
    [u_1*iSP LCR_1 u_1*Temp_1 u_1*iType_1];  
    [u_2*iSP LCR_2 u_2*Temp_2 u_2*iType_2]  
    [...];  
];
```

and so on.

Of course, these boundaries can also be read from a database. This way a PhD student reads in 635000 lines at ones and transfers these into a boundary list for input.

## 5.1 Constant concentration cells cannot be switched off, **helas!!**

The MT3DMS manual states for the SSM package that constant concentration cells ITYPE=1 cannot be switched off in subsequent stress periods once specified. It is possible to change the concentrations, however. From the point of usage this is a bad feature, because it is not possible to create an initial situation in one stress period and let it die out in the next periods.

Such situation can, of course be computed using an intermediate step, i.e. first run a model to generate the wanted situation. Use those concentration as the start heads of the next run

# 6 Understanding Seawat input for viscosity and density

The input for the VDF and VSC modules in Seawat are flexible but terribly difficult to comprehend as result of the possible switches. After having spent in total several days wrestling with it, I attempted to make the description more easy to understand. Nevertheless, I hope that this input will be severely overhauled in the future so that people don't have to waste part of their remaining life time trying to figure out the tweaks of this way of specifying this input. I'm convinced it can be done more rigorously and straightforward as it still has some inconsistencies, especially with the options to read in density or viscosity data for specific stress periods and on the same time using the multi-species capabilities. These two are not compatible given the input structure.

One way is to include the logic-scheme of the input instructions (figure 11).

## 6.1 Boundary conditions for constant head with variable density

Variable density boundary conditions can be somewhat complicated especially when the density changes during a simulation. The Seawat V4 manual on page

12-14 provides a clear explanation of the complexities and how to deal with them using the options provided by Seawat V4. The authors favor using CHD boundary package over ICBUND for given concentrations because CHD boundaries can vary during the simulation, for instance because of density changes. Instructions are given in on page 22. It's usage can be found in the mf\_adapt of the [examples/swt\\_V4/Coast](#).

To make the CHD package aware of the CHDDENSOPT it must be specified as a variable in mf\_adapt like

```
CHDDENSOPT=2; % use environmental head at ocean boundary,
Langevin et al 2008, p22
```

The value doesn't matter per se for the CHD package, but it can elegantly be used in the specification of the CHD input column where the CHDDENSOPT values has to be specified see below (6th column). If CHDDENSOPT is 1, an extra field CHDDENS is required. This can be done in the same way. Specify the variable and add its value as the right most (7th) column of CHD input.

```
..
LRCright=cellIndices(find(XM>xGr(end-1)),size(M),'LRC');

CHD=[];...

for iPer=1:NPER

    CHD=[CHD;

        [iPer*u LRCright u*[h_ocean h_ocean CHDDENSOPT]]

    ];

end
```

## 7 Steady-state versus transient flow with transport

One feature that often causes confusion is steady state of the flow model versus steady state of the transport model MT3MDS or SEAWAT. Even though the flow model maybe steady state, the transport model remains transient. Therefore, the time specified in the stress period for steady state periods matters for as far as the MT3DMS or SEAWAT are concerned. However, in case of a steady state stress period, the steps specified within that period don't matter. The flow model will compute the steady-state solution in a single step, whereas the transport model steps through time at the pace of its own transport steps, which are determined by the maximum permissible step size.

## 7.1 Viscosity in the NAM file with density package off

To use the viscosity package Seawat must run. But one may want to use viscosity without the density package on. mFLab is triggered to generate the input for Seawat, when it sees that the VDF package in the NAM worksheet is “on”. Specifically to run Seawat without the density package on one may specify the on-switch for the VDF package on the NAM sheet as -1 instead of 1.

## 7.2 Density package

Figure 10 shows a mindmap of the input instructions of the Seawat V4 manual.

### 7.2.1 MT3DRHOFLAG ( $\rho Flag$ )

$\rho Flag$  is the major 3-way switch in the density package. It can be -1, or  $>-1$  (i.e.  $\geq 0$ ).

**if  $\rho Flag \geq 0$**  If  $\rho Flag \geq 0$  then either the density is read in per stress period or it is computed with only one MT3DMS species is involved:

$$\rho = \rho_R + \frac{\partial \rho}{\partial c} c$$

There is no reference concentration included, which, therefore implies it is taken to be zero in Seawat if computed using item 4), where only  $\rho_R$  and  $\frac{\partial \rho}{\partial c}$  are specified and no reference concentration  $c_R$  as is required in item 4c (see 5).

The manual says that if  $\rho flag > 0$  it is the MT3DMS species number, however if it is zero, no MT3DMS species number is used or at least required by Seawat, as the density will be read in directly or through its concentration (4). It is not clear if and if yes which species number Seawat uses in case  $\rho Flag = 0$ .

**$\rho Flag = 0$  (reading density or concentration for each stress period)** if  $\rho Flag = 0$ , then non concentration species is involved and density will be read in or specified for each stress period according to the flag INDENSE.. This means that densities may be read for some stress periods while they may be computed for other stress periods. This flag INDENSE works as follows:

If INDENSE $<0$ , the data from the previous period are reused or DENSEREF if the first stress period.

If INDENSE=0, set all to DENSEREF

if INDENSE  $>0$ , read item 7 (DENSE or CONCENTRATIONS) for that stress period.

if INDENSE=2, concentrations are read and converted to densities internally.

Directly reading of cell-density values will be rare. Its most likely application is a restart from a previous run.

- Items that are needed per stress period are specified in *mfLab* in the PER worksheet column “INDENSE” of the workbook for the problem on hand. If INDENSE is 1 for a stress period, then *mfLab* expects to find the specification of the densities to be read in the workspace parameter DENSE which must be a cell array with the cell corresponding to the stress period for which INDENSE==1 holding the 3D array with density values for all cells of the model.

$\rho Flag < 0$ , ( $\rho Flag = -1$ ) **density computed using any series of species**  
If  $\rho Flag < 0$ , Seawat will compute density using NSRhoEOS (zero or more) species with a *linear* relation

$$\rho = \rho_R + \frac{\partial \rho}{\partial c} (c - c_{\rho_R}), \quad \left\{ \rho_{ref}, \frac{\partial \rho}{\partial c}, c_{\rho_R} \right\} \quad (4)$$

Item 4c) then reads the parameter for the linear relations

$$\left\{ k_i, \frac{\partial \rho}{\partial c_k}, c_{k, \rho_R} \right\}_{i=1 \dots NSRhoEOS} \quad (5)$$

where

- $i$  = the number in the list  $1 \dots NSRhoEOS$
- $k_i$  = the MT3DMS species number for this relation
- $c_k$  = the concentration of this species
- $c_{k, \rho_R}$  = the concentration of this species when the water has its reference density

Note that the reference density, DENSEREF, itself is the same for all species and read in separately in item 4a). This is done together with parameters that specify the relation between density and pressure head  $\frac{\partial \rho}{\partial \phi_p} \simeq 4.46 \times 10^{-3} \text{ kg/m}^4$  in terms of the reference density:

$$\Delta \rho_P = \frac{\partial \rho}{\partial \phi_p} (\phi_p - \phi_{p_R})$$

Clearly, NSRhoEOS >= 0, otherwise no species are available to compute the density.

## 8 Viscosity package

Figure 11 shows a mindmap of the input instructions of the Seawat V4 manual.

### 8.0.2 MT3DMUFLAG ( $\mu Flag$ )

$\mu Flag$  is the major 3-way switch in the viscosity package. It can be -1, or >-1 (i.e.  $\geq 0$ ).



$\mu Flag \geq 0$  If  $\mu Flag \geq 0$  then then only 1 MT3DMS species is involved in the viscosity computation. And it is obliged to specify for this species the three parameters needed for a linear computation of the relation between viscosity and this species' concentration

$$\mu = \mu_{ref} + \frac{\partial \mu}{\partial c} (c - c_{\mu_{Ref}}), \quad \left\{ \mu_{ref}, \frac{\partial \mu}{\partial c}, c_{\mu_{Ref}} \right\} \quad (6)$$

This shows that any species can be used in this way to compute viscosity linearly, including but not necessarily, temperature.

The manual says that  $\mu flag$  is the MT3DMS species number, but this conflicts with  $\mu Flag = 0$ , being an illegal species number.

if  $\mu Flag > 0$ , then  $\mu Flag$  is the MT3DMS species number used for the concentration in (4).

$\mu Flag = 0$  if  $\mu Flag = 0$ , then viscosity will be read in for each stress period, but only if  $INVISC > 0$  (item 4) for that stress period.

This implies that we can still have stress periods with  $INVISC = 0$  and at the same time  $\mu Flag = 0$ , so that then Seawat may only compute viscosity using the parameter specified by (4) in item 3, without a species number being specified. From a user's perspective it is unclear how Seawat does this, without involving any MT3DMS species or using some MT3DMS default species.

Directly reading of cell-viscosity values will be seldom. It's most likely application is a restart from a previous run. In that case, one may as well read in temperature or related species directly instead of viscosity.

- Items that are needed per stress period are specified in *mfLab* in the PER worksheet column "INVISC" of the workbook for the problem on hand. If INVISC is 1 for a stress period, then *mfLab* expects to find the specification of the viscosities to be read in the workspace parameter VISC which must be a cell array with the cell corresponding to the stress period for which INVISC==1 holding the 3D array with viscosity values for all cells of the model.

$\mu Flag < 0$ , ( $\mu Flag = -1$ )

If  $\mu Flag < 0$ , Seawat will compute viscosity using NSMUEOS (zero or more) species with a *linear* relation and, optionally and additionally to NSMUEOS, by a non-linear relation between temperature and viscosity.

Item 3d then reads the parameter for the linear relations

$$\left\{ k_i, \frac{\partial \mu}{\partial c_k}, c_{k, \mu_{Ref}} \right\}_{i=1 \dots NSMUEOS} \quad (7)$$

where

$i$  = the number in the list  $1 \dots NSMUEOS$

$k_i$  = the MT3DMS species number for this relation



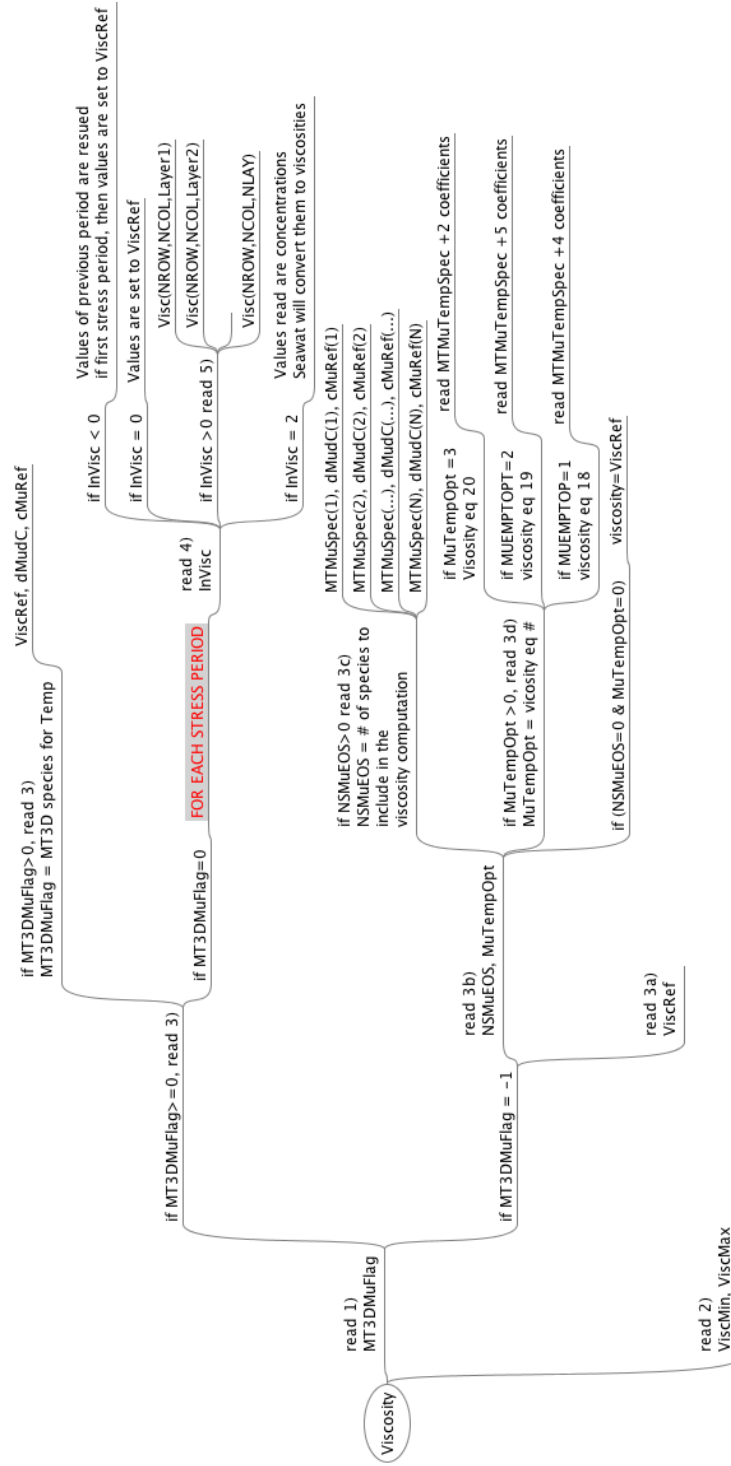


Figure 11: Viscosity input scheme SEAWAT V4

$c_k$  = the concentration of this species

$c_{k,\mu_{Ref}}$  = the concentration of this species when the water has its reference viscosity

Note that the reference viscosity,  $VISCREF$ , itself is the same for all species and read in separately in item 3a).

Clearly, temperature may be one of the species just specified, but then it can only have a linear relation with viscosity. This is not generally sufficient. Therefore, the NSMUEOS species for linear relations are most suitable for the relation between viscosity and the concentration of certain species that affect it measurably.

It is also clear that  $NSMUEOS=0$  is acceptable, as it means that no species affects viscosity in a linear fashion.

The relation between temperature and viscosity is specified using the  $MUTEMP-TOPT$  flag that is read in together with  $NSMUEOS$ .

### 8.0.3 $MUTEMP OPT$ ( $\mu$ temperature option)

$MUTEMP OPT$  is read in together with  $NSMUEOS$  in item 3b).  $MUTEMP OPT$  can be 0, 1, 2 or 3. If it is 0 and  $NSMUEOS=0$  then the viscosity is fixed to  $VISCREF=\mu_{Ref}$  in the entire model. If it is 1, 2 or 3 Seawater will compute the viscosity using a non-linear relation with temperature, specified in equation 18, 19 en 20 of the manual, on page 6.

Each of these equations has its own set of parameters (2, 5, and 4 respectively), which has to be specified in the input, headed by the  $MT3DMS$  species that is used for the temperature. This is done in item 3).

Note that this non-linear temperature relation is specified completely separated from the species involved in  $NSMUEOS$ . Therefore, the species number  $MTMUTEMP SPEC$  (see 5) must be different from any of the species numbers specified under  $NSMUEOS$  in item 3c) and it must be the species holding the temperature.