Fixed-gradient boundary condition in Modflow

Theo Olsthoorn

6 Nov 2014

1 Introduction

Models dealing with groundater in desert enviroments generally lack clear head boundaries to fixed their levels, except where their water table is shallow and, therefore, affacted by evopration and or transpiration. Groundwater models, however should be provided logical boundaries where in reality such boundaries do not exist. For instance, fixed-head boundaries are generally completely irrealistic under such conditions. However, even to a lesser extent, the same holds true for fixed-flow conditions. Even Cauchy boundary conditions are not logical as they are a mix of both fixed-head and fixed-flow conditions and, therefore, not logical in a desert enviroment.

A compromize allowing the flow across model boundaries to be adapted to coditions that change within the model, is using fixed-gradient boundary conditions. This allows the water table to fluctuate, while the flow is adapted to the change of the transmissivity caused by the change of the water table elevation. This too, of course is a compromize, but may be a more realistic one than either a fixed head, a fixed flow or a Cauchy boundary condition.

2 Derivation

2.0.1 Fixed gradient boundary at the downstream side of the model

Achieving a fixed-gradient boundary condition a tthe downstream side of a finite difference model like Modflow. Let the gradient at the desired boundary be $i\,[\mathrm{L/L}]$ then the flow Q across a line of width B perpendicular to it equals

$$Q\left(h\right) = iBT\left(h\right)$$

with T the transmissivity. B may be considered the cell width. The groundwater head gradient, i, is supposed to be constant, but the water table is not; both Q and T are thus functions of the elevation of the water table, h.

We may now write

$$\begin{aligned} \frac{\partial Q}{\partial h} &= iB\frac{\partial T}{\partial h} \\ &= iBk_h \end{aligned}$$

We may integrate this from the bottom, h_0 of the aquifer to the eactual water table elevation h

$$Q_h = iB\int_{h_0}^{h} k dh = iB\left(T_h - T_{h_0}\right)$$

If we set $Q_0 = 0$ to correspondwith $h = h_0 = z_0$, the base of the aquifer, and because the concutivity is piecewise constant, as it correponds with the discrete layers of the model, we have with index l referring to layer number that is counted from the bottom upward

$$\Delta Q = iB\sum_{l=1}^{n-1} k_l \Delta z_l + iB\left(h - z_n\right)k_n \tag{2.1}$$

where n is the layer number with the water table in it and z_n the elevation of the bottom of layer n.

We may implement this in MODFLOW by applying its drain boundary condition, using a "drain" located at the bottom of each layer at this boundary. The drain elevation is then fixed, but we still need to define its conductance.

A drain at the bottom of every boundary cell implies that each model layer will start contributing to the discharge only when the water table is abov the the bottom of that

layer. It does so proportionally with the head relative to the elevation of the drain, i.e. relative to the bottom of the layer.

When the water table is at some level, the discharge according to equation **??** must thus be matched by the drains installed below it. To this end write the right-hand side of equation 2.1. Leaving out for writing convenience the factor iB, we get

$$\begin{aligned}\sum_{i=1}^{n-1} k_i \Delta z_i + k_n (h - z_n) &= \sum_{i=1}^{n} k_i (h - z_i) - \sum_{i=2}^{n} k_{i-1} (h - z_i) \\ &= k_1 (h - z_1) + k_2 (h - z_2) + ...k_n (h - z_n) ... - k_1 (h - z_2) - k_2 (h - z_3) + ...k_{n-1} \\ &= .k_1 (z_2 - z_1) + k_2 (z_3 - z_2) + ...k_{n-1} (z_n - z_{n-1}) ...k_n (h - z_n) -\end{aligned}$$

which is correct. But to use DRN boundaries, we should order the terms differently as follows

$$\begin{aligned}\sum_{i=1}^{n-1} k_i \Delta z_i + k_n (h - z_n) &= \sum_{i=1}^{n} k_i (h - z_i) - \sum_{i=2}^{n} k_{i-1} (h - z_i) \\ &= k_1 (h - z_1) + (k_2 - k_1) (h - z_2) + (k_3 - k_2) (h - z_3) ... + (k_n - k_{n-1}) (h - z_n)\end{aligned}$$

which is a set of MODFLOW drains as was described above. The only thing to to is to supply the correct conductances, which is

$$\begin{aligned}C_1 &= iBk_1 \\ C_2 &= iB (k_2 - k_1) \\ C_3 &= iB (k_3 - k_2) \\ ... &= ... \\ C_n &= iB (k_n - k_{n-1})\end{aligned}$$

Therefore, when we implement a drain in each cell with their elevaton equal to that of the bottom of the cells and a proper conductance as given. The discharge will then match the constant gradient boundary condition exactly. The conductance is to be computed as

$$\begin{aligned}C_1 &= iB (k_1 - 0) \\ C_i &= iB (k_i - k_{i-1})\end{aligned}$$

$Bk_j^* \Delta h_j$ in which k_j^* is computed as

$$k^* = [k_1 \ \text{diff}(k)]$$

in which k_1 is the conductivity of the lowest cell and diff(k) is the difference between successif conductivities in the column from the bottom upward. The value of $Q_0 = 0$ and, so it does not have to be considered. The easiest way to implement a fixed gradient boundary would be to write a specific function for it or implement it as a method of line objects

2.0.2 Fixed gradient boundary at the upstream side of the model

The flow at the upstream side of the model given its fixed gradient is achieved in the same way as for the downstream side of a model. The only thing that needs to be done is to change the sign of the gradient, i.e. negative at the upstream side from which the water flows into the model and positive at the downstream side. A negative gradient is equivalant to changing the sign of the conductances. What happens is that, when the head is above the drain elevation, water starts flowing into the model due to negative conductance. This implies that inflow increases with increasing water levels, which is what was to be achieved.

2.1 Example

The fixed gradient boundary condition is demonstrated in the *mfLab* example *fixedGrad-BCN.*.

To easy its usa in a practical model, it was implemented in a method of *lineObj*, *lineObj.setGradient(gr,gradient,HK)*. Each such *lineObj* will be copied *Nlay* times and its properties adapted so that it changes the necessary drains at thte bottom of the layers along this *lineObj*.