16

Solving the 1D Richards Equation by Finite-Difference Approximation

Start with continuity:

$$\frac{\partial \theta}{\partial t} = C(h)\frac{\partial h}{\partial t} = -\frac{\partial q}{\partial z} = -\frac{\partial}{\partial z}\left[-K\frac{\partial h}{\partial z} - K\right] = \frac{\partial}{\partial z}\left[K\frac{\partial h}{\partial z}\right] + \frac{\partial K}{\partial z}$$

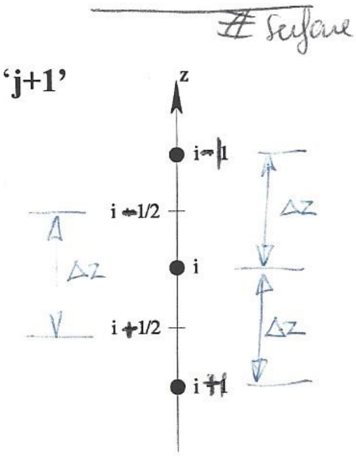
For vertical flux (z positive upwards), Darcy gives:

$$C(h)\frac{\partial h}{\partial t} = \frac{\partial}{\partial z}\left[K(h)\left(\frac{\partial h}{\partial z} + 1\right)\right] = \frac{\partial}{\partial z}\left[K(h)\frac{\partial h}{\partial z}\right] + K'(h)\frac{\partial h}{\partial z}$$

Introduce finite-difference approximation ($\partial \rightarrow \Delta$):

$$C(h)\frac{\Delta h}{\Delta t} = -\frac{\Delta q}{\Delta z}$$

For numerical stability, C and q will be evaluated at the **end of the time step**, i.e., at **'j+1'** ("fully implicit method"). Use notation $C_{i,j+1}$ for $C(h_{i,j+1})$ etc.



$$C_{i,j+1}\frac{h_{i,j+1} - h_{i,j}}{\underset{>0}{\Delta t}} = -\frac{q_{i+½,j+1} - q_{i-½,j+1}}{\underset{<0}{\Delta z}}$$

if z>0 upward.

Substitute Darcy, define $K' = \frac{\partial K}{\partial h}$ (*analytically* obtained):

$$C_{i,j+1}\frac{h_{i,j+1} - h_{i,j}}{\Delta t} = \frac{K_{i+½,j+1}\frac{h_{i+1,j+1} - h_{i,j+1}}{\Delta z} - K_{i-½,j+1}\frac{h_{i,j+1} - h_{i-1,j+1}}{\Delta z}}{\Delta z}$$

$$+ K'_{i,j+1}\frac{h_{i+1,j+1} - h_{i-1,j+1}}{2\Delta z}$$

j+1

Expand $K_{i+½,j+1}$ and $K_{i-½,j+1}$:

$$C_{i,j+1}\frac{h_{i,j+1} - h_{i,j}}{\Delta t} = \frac{\frac{K_{i,j+1} + K_{i+1,j+1}}{2}\frac{h_{i+1,j+1} - h_{i,j+1}}{\Delta z} - \frac{K_{i-1,j+1} + K_{i,j+1}}{2}\frac{h_{i,j+1} - h_{i-1,j+1}}{\Delta z}}{\Delta z}$$

$$+ K'_{i,j+1}\frac{h_{i+1,j+1} - h_{i-1,j+1}}{2\Delta z}$$

$h_{i-1,j+1}$

Now re-arrange:

$$h_{i,j}\frac{-C_{i,j+1}}{\Delta t} = h_{i-1,j+1}\left(\frac{K_{i-1,j+1}+K_{i,j+1}}{2(\Delta z)^2} - \frac{K'_{i,j+1}}{2\Delta z}\right) +$$

$$h_{i,j+1}\left(-\frac{K_{i,j+1}+K_{i+1,j+1}}{2(\Delta z)^2} - \frac{K_{i-1,j+1}+K_{i,j+1}}{2(\Delta z)^2} - \frac{C_{i,j+1}}{\Delta t}\right) +$$

$$h_{i+1,j+1}\left(\frac{K_{i,j+1}+K_{i+1,j+1}}{2(\Delta z)^2} + \frac{K'_{i,j+1}}{2\Delta z}\right)$$

Defining

$$a_i = \left(\frac{K_{i-1,j+1}+K_{i,j+1}}{2(\Delta z)^2} - \frac{K'_{i,j+1}}{2\Delta z}\right),$$

$$b_i = \left(-\frac{K_{i,j+1}+K_{i+1,j+1}}{2(\Delta z)^2} - \frac{K_{i-1,j+1}+K_{i,j+1}}{2(\Delta z)^2} - \frac{C_{i,j+1}}{\Delta t}\right),$$

$$c_i = \left(\frac{K_{i,j+1}+K_{i+1,j+1}}{2(\Delta z)^2} + \frac{K'_{i,j+1}}{2\Delta z}\right),$$

$f_i = h_{ij}\frac{-C_{ij+1}}{\Delta t} = h_{ij} * d_i$

$\cancel{f_i = h_{i,j}\frac{-C_{i,j+1}}{\Delta t}}$, $h_{ij} * d_i$, where $d_i = -\frac{C_{ij+1}}{\Delta t}$

we can write the equation for an *internal* node 'i' as follows:

$$a_i h_{i-1,j+1} + b_i h_{i,j+1} + c_i h_{i+1,j+1} = f_i \qquad [16]$$

Thus, for this type of approximation, the h-value at spatial node 'i' at time 'j+1' depends only on the two nearest neighbors, 'i-1' and 'i+1,' as well as on $h_{i,j}$. In matrix notation, we have for a five-node system:

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & 0 \\ 0 & 0 & a_4 & b_4 & c_4 \\ 0 & 0 & 0 & a_5 & b_5 \end{bmatrix} \underbrace{\begin{bmatrix} h_{1,j+1} \\ h_{2,j+1} \\ h_{3,j+1} \\ h_{4,j+1} \\ h_{5,j+1} \end{bmatrix}}_{\Uparrow \ \textbf{UNKNOWN}} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix}_j$$

Obviously, b_1, c_1, f_1 and a_5, b_5, f_5 have to obtained from the *boundary conditions* of the particular problem.

And rewrite { matrix equations for $i=2$ and 4:
using [16] ← side aside

$\underline{\underline{i=2}}$:

$$a_2 h_{1,j+1} + b_2 h_{2,j+1} + c_2 h_{3,j+1} = f_2$$

or $$b_2 h_{2,j+1} + c_2 h_{3,j+1} = f_2 - a_2 h_{1,j+1}$$

known boundary conditions.

$\underline{\underline{i=4}}$:

$$a_4 h_{3,j+1} + b_4 h_{4,j+1} = f_4 - c_4 h_{5,j+1}$$

Yields.

known

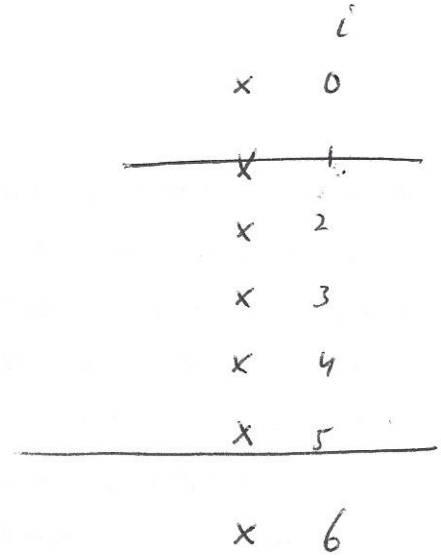
$$\begin{bmatrix} b_2 & c_2 & 0 \\ a_3 & b_3 & c_3 \\ 0 & a_4 & b_4 \end{bmatrix} \begin{bmatrix} h_2 \\ h_3 \\ h_4 \end{bmatrix}_{j+1} = \begin{bmatrix} f_2 - a_2 \overset{\text{UTOP}}{h_{1,j+1}} \\ f_3 \\ f_4 - c_4 \overset{\text{UBOT}}{h_{5,j+1}} \end{bmatrix} \quad [17]$$

known

Thus if h boundary conditions at both ends
then $[N-2]$ equations and unknowns,
where $\underline{N \text{ is total \# of nodes}}$. [Dirichlet boundary condition]

However, alternatively one can define
flux boundary condition, i.e., $q = -K(h)\dfrac{dH}{dz}$.
$\Rightarrow$ Neumann b.c

Flux b.c.: [Neumann b.c.]



Assign fictitious (false) gridpoints: h_0 and h_6.

From [16]

$$a_i\, h_{i-1,j+1} + b_i\, h_{i,j+1} + c_i\, h_{i+1,j+1} = d_i\, h_{ij}$$

$$a_1\, h_{0,j+1} + b_1\, h_{1,j+1} + c_1\, h_{2,j+1} = d_1\, h_{1,j} \qquad i=1$$

$$\vdots \qquad\qquad [16]$$

$$a_5\, h_{4,j+1} + b_5\, h_{5,j+1} + c_5\, h_{6,j+1} = d_5\, h_{5,j} \qquad i=5$$

Now 7 unknown's [h_0 through h_6] and 5 equation's

Darcy equation: for $i=1$

$$q_{1,j+1} = -K_{1,j+1}\left[\underbrace{\frac{h_{2,j+1}-h_{0,j+1}}{2\Delta z}}_{\text{central difference approx.}} + 1\right]$$

$$\frac{q_{1,j+1}}{K_{1,j+1}} + 1 = -\left(\frac{h_{2,j+1}-h_{0,j+1}}{2\Delta z}\right)$$

or: $$h_{0,j+1} - h_{2,j+1} = 2\Delta z\left[\frac{q_{1,j+1}}{K_{1,j+1}} + 1\right] = FLUT$$

(is known from flux b.c.)

$$\Longrightarrow \boxed{h_{0,j+1} = h_{2,j+1} + FLUT}$$

thus, h_0 is determined such that flux through surface is equal to prescribed flux b.c.

i.e. Check for $q_1 = 0$

then $h_0 = h_2 + 2\Delta z \longrightarrow$ yields $q_1 = \phi$ ok.

Substitute result in [16] for $i=1$

$$a_1 \left[h_{2\,j+1} + FLUT \right] + b_1 h_{1\,j+1} + c_1 h_{2\,j+1} = d_1 h_{1j}$$

or: $\Rightarrow$ $$b_1 h_{1\,j+1} + \left[a_1 + c_1 \right] h_{2\,j+1} = d_1 h_{1j} - a_1 * FLUT. \quad [17]$$

set aside

Similarly for bottom with $i=5$

$$q_{5\,j+1} = -K_{5\,j+1} \left[\frac{h_{6\,j+1} - h_{4\,j+1}}{2\Delta z} + 1 \right]$$

$$h_{6\,j+1} - h_{4\,j+1} = -2\Delta z \left[\frac{q_{5\,j+1}}{K_{5\,j+1}} + 1 \right] = FLUB \leftarrow$$

$$\underline{h_{6\,j+1} = h_{4\,j+1} + FLUB.}$$

Substitute in [16] for $i=5$.

$$a_5 h_{4\,j+1} + b_5 h_{5\,j+1} + c_5 \left[h_{4\,j+1} + FLUB \right] = d_5 h_{5j}$$

or $$\left[a_5 + c_5 \right] h_{4\,j+1} + b_5 h_{5\,j+1} = d_5 h_{5j} - c_5 FLUB. \quad [18]$$

set aside

Now new matrix solution

$$\begin{bmatrix} b_1 & (a_1+c_1) & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & 0 \\ 0 & 0 & a_4 & b_4 & c_4 \\ 0 & 0 & 0 & (a_5+c_5) & b_5 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{bmatrix}_{j+1} = \begin{bmatrix} d_1 h_1 - a_1 FLUT \\ d_2 h_2 \\ d_3 h_3 \\ d_4 h_4 \\ d_5 h_5 - c_5 FLUB \end{bmatrix}_j \quad [19]$$

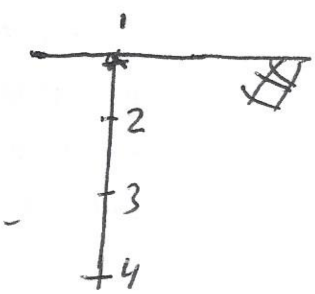
(false)

but 5 rows

Fictitious grids is convenient way to represent flux bc, but not required. Alternatively, use Taylor series expansion to express h_1 or h_5 from 2 inner nodes and flux b.c. $\Longrightarrow$ 3 point - finite difference approx.

See Vemuri & Karplus, 1988; page 96.

Expand both h_2 and $_3$ around h_1



$$4 \cdot h_2 = 4 \cdot h_1 + 4 \cdot \left.\frac{dh}{dz}\right|_1 \Delta z + 4 \cdot \left.\frac{d^2h}{dz^2}\right|_1 \frac{(\Delta z)^2}{2} + \cdots$$

$$h_3 = h_1 + \left.\frac{dh}{dz}\right|_1 (2\Delta z) + \left.\frac{d^2h}{dz^2}\right|_1 \frac{(2\Delta z)^2}{2} + \cdots$$

$-$

Subtract

Use [19] for both Neumann (flux) and Dirichlet (head) boundary condition.

Allow [19] for head b.c. :

Set: $b_1 = 1$

$a_1 + c_1 = \emptyset$

$d_1 h_1 - a_1 FLUT = UTOP$

$a_2 = \emptyset$

Set bottom row:

$a_5 + c_5 = \emptyset$

$b_5 = 1$

$d_5 h_5 - c_5 FLUB = UBOT$

$c_4 = \emptyset$

Now: same matrix as Eq [17]

NEUMANN Single Matrix for **both** boundary conditions

$$\begin{bmatrix} b_1 & (a_1+c_1) & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & 0 \\ 0 & 0 & a_4 & b_4 & c_4 \\ 0 & 0 & 0 & (a_5+c_5) & b_5 \end{bmatrix} * \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{bmatrix}_{j+1} = \begin{bmatrix} d_1 h_1 - a_1\, FLUT \\ f_2 \\ f_3 \\ f_4 \\ d_5 h_5 - c_5\, FLUB \end{bmatrix}_j$$

DIRICHLET

$$\begin{bmatrix} 1 & \cancel{0} & 0 & 0 & 0 \\ 0 & b_2 & c_2 & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & 0 \\ 0 & 0 & a_4 & b_4 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{bmatrix}_{j+1} = \begin{bmatrix} UTOP \\ f_2 - a_2\, UTOP \\ f_3 \\ f_4 - c_4\, UBOT \\ UBOT \end{bmatrix}_j$$

From:

$$a_i h_{i-1,j+1} + b_i h_{i,j+1} + c_i h_{i+1,j+1} = f_i \qquad [16]$$

$i = 1, \ldots, 5$

$a_1 = 0$
$c_5 = 0$

Tridiagonal Algorithm

$$\left.\begin{array}{l} Ax = b \\ A = L \cdot U \end{array}\right\} \quad L.U.x = b$$

$$A = \begin{bmatrix} b_1 & c_1 & & \\ a_2 & b_2 & c_2 & 0 \\ & \ddots & \ddots & \ddots \\ 0 & & a_n & b_n \end{bmatrix}$$

$L.U = A$

solve to get α_i and β_i of L and U.

$$\Downarrow$$

$$L = \begin{bmatrix} \alpha_1 & & & \\ a_2 & \alpha_2 & & 0 \\ & \ddots & \ddots & \\ 0 & & a_n & \alpha_n \end{bmatrix}$$

$$U = \begin{bmatrix} 1 & \beta_1 & & \\ & 1 & \beta_2 & 0 \\ & & \ddots & \ddots \\ & & & \beta_{n-1} \\ 0 & & & 1 \end{bmatrix}$$

$\alpha_1 * 1 = b_1 \longrightarrow \alpha_1 = b_1$

$\alpha_1 * \beta_1 = c_1 \longrightarrow \beta_1 = c_1 / \alpha_1$

$a_2 * \beta_1 + \alpha_2 * 1 = b_2 \rightarrow \alpha_2 = \dots$

$a_2 * 0 + \alpha_2 * \beta_2 = c_2 \rightarrow \beta_2 = \dots$

<u>In General:</u>

$\alpha_1 = b_1$ and $\beta_1 = c_1/\alpha_1$

$$\alpha_i = b_i - a_i * \beta_{i-1}$$

$$\beta_i = c_i / \alpha_i$$

for $i = 2, \dots, N$.

Tridiagonal or Thomas Algorithm.

Solve $Ax = b$, where A = tri diagonal.
and b is known.
Solve for x!

Set $A = L.U$ where L: Lower diagonal matrix
U: Upper diagonal matrix.

$L.U\,x = b$ Set $y = Ux$ and $\boxed{\text{solve } Ly = b.}$

Solve for $[y]$ first, then solve for $[x]$.

$$\begin{bmatrix} \alpha_1 & & & & 0 \\ a_2 & \alpha_2 & & & \\ 0 & \ddots & \ddots & & \\ & & & & \\ 0 & & & a_n & \alpha_n \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ \\ \\ y_n \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ \\ \\ b_n \end{bmatrix}$$

$\alpha_1 y_1 = b_1 \rightarrow y_1 = \frac{b_1}{\alpha_1}$

$a_2 y_1 + \alpha_2 y_2 = b_2 \longrightarrow$

$y_2 = \frac{b_2 - a_2 y_1}{\alpha_2}$ or

$$\boxed{y_i = \frac{b_i - a_i y_{i-1}}{\alpha_i}}$$

forward

Then $\boxed{\text{solve for } Ux = y}$

$$\begin{bmatrix} 1 & \beta_1 & & & \\ & 1 & \beta_2 & & \\ & & \ddots & \ddots & \\ & 0 & & \ddots & \beta_{n-1} \\ & & & & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ \\ \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ \\ \\ y_n \end{bmatrix}$$

backward substitution:

$$\boxed{x_n = y_n}$$

$x_{n-1} = y_{n-1} - \beta_{n-1} x_n$

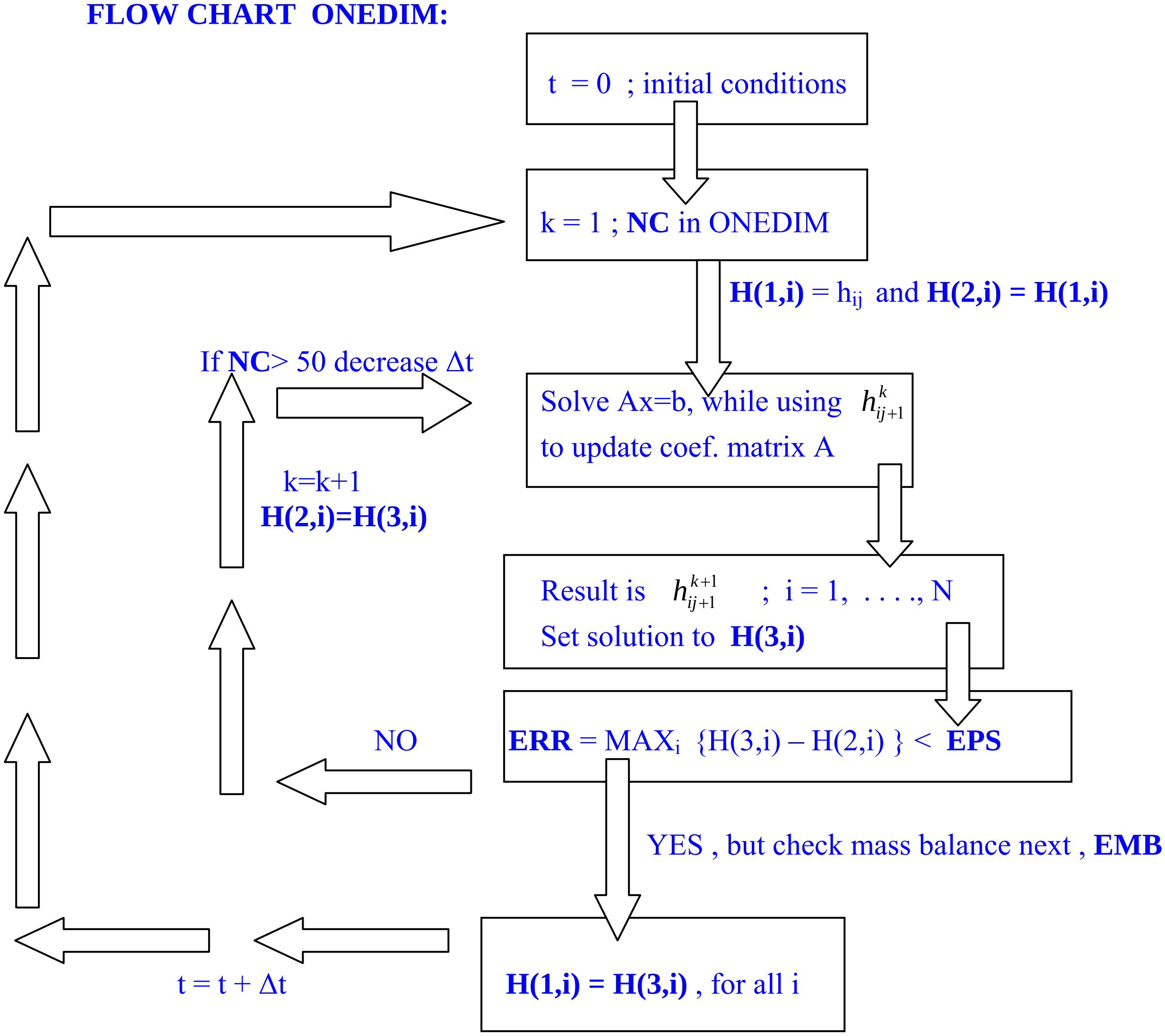
$$\boxed{\begin{array}{c} x_i = y_i - \beta_i * x_{i+1} \\ i = (n-1), \ldots, 1 \end{array}}$$

Richards' Equation Flow Charts and Mass Balance Calculations:

In solving Eq. [21], a_i, b_i, and c_i, that include C_i, K_i, as a function of h, are at (j+1) time level. So, although they are in the coefficient matrix, their true values at (j+1) are unknown. How to solve???

At beginning of new time step, t + Δt, values of K_{ij} and C_{ij} are substituted for K_{ij+1} and C_{ij+1}. The coefficient matrix is solved iteratively, until differences between h_{ij+1}^{k+1} and h_{ij+1}^{k} are smaller than error criterion, ε (EPS) . Realize that superscript (k) denotes the iteration number. This method is called the PICARD ITERATION.

FLOW CHART ONEDIM:



Define relative error between iterations as:

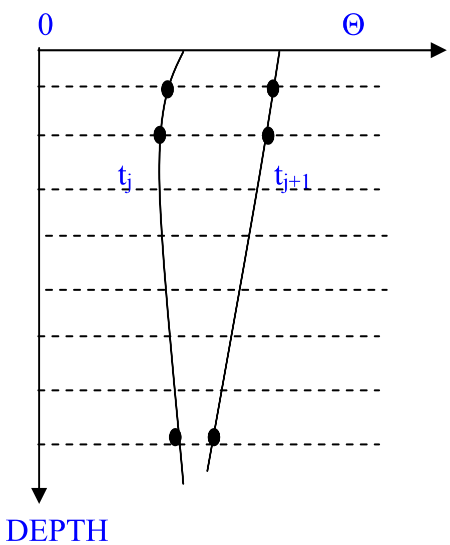
$$ERR = \max_{i=1,..,N} \left| \frac{h_{ij+1}^{(k+1)} - h_{ij+1}^{k}}{h_{ij+1}^{(k+1)}} \right|$$

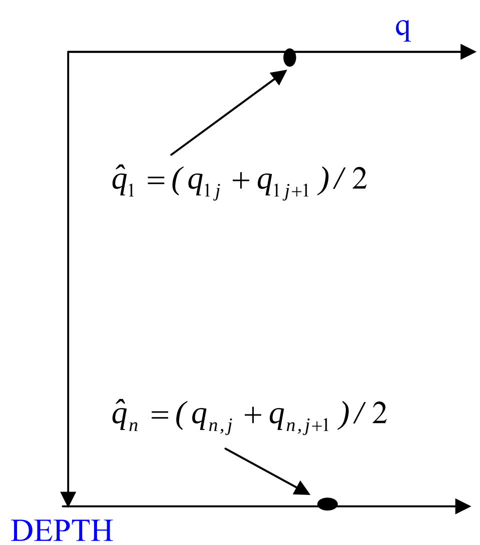
If ERR > ε , Re-iterate
If ERR < ε , OK, and h_{ij+1}^{k+1} is the solution. In ONEDIM, ε = ERR = 0.001

Program ONEDIM limits the nr of iterations NC to 50.
If NC > 50, reduce time step by one half, and start over.

So, how about time step size Δt = DT (in ONEDIM):
This is done using mass balance calculations:

MASS BALANCE CALCUATIONS in ONEDIM





$\frac{\partial \theta}{\partial t} = -\frac{\partial q}{\partial z}$, for the complete soil domain. That is, between the surface and bottom boundary.

$$d\theta\ dz = -dq\ \partial t \qquad or \qquad \int \Delta\theta dz = -\int \Delta q\ dt$$

$$\Delta\theta = \theta_{i,j+1} - \theta_{i,j} \qquad and \qquad \Delta q = q_{in} - q_{out} = \hat{q}_1 - \hat{q}_n$$

DELMO = $\int \Delta\theta \, dz = \int_{DEPTH}^{0} (\theta_{i,j+1} - \theta_{i,j})dz$ DELFLU = $-\int_{t_j}^{t_{j+1}} (q_{in} - q_{out})dt$

SO:

Absolute mass balance, EMB = DELMO – DELFLU
Relative mass balance, REMB = EMB/DELFLU

Now:
If EMB > DEL (DEL = 0.001)
Decrease Δt: Δt = 0.5Δt
Start over again, i.e., reject $h_{i,j+1}$ and start re-iterating.

IF EMB < (0.1xDEL)
Increase Δt: Δt =1.5Δt
Now advance in time.

IF 0.1DEL < EMB < DEL
Advance in time,
But do not change Δt