Lecture Modelling of Hydro-systems

Mass Transport Process Part II

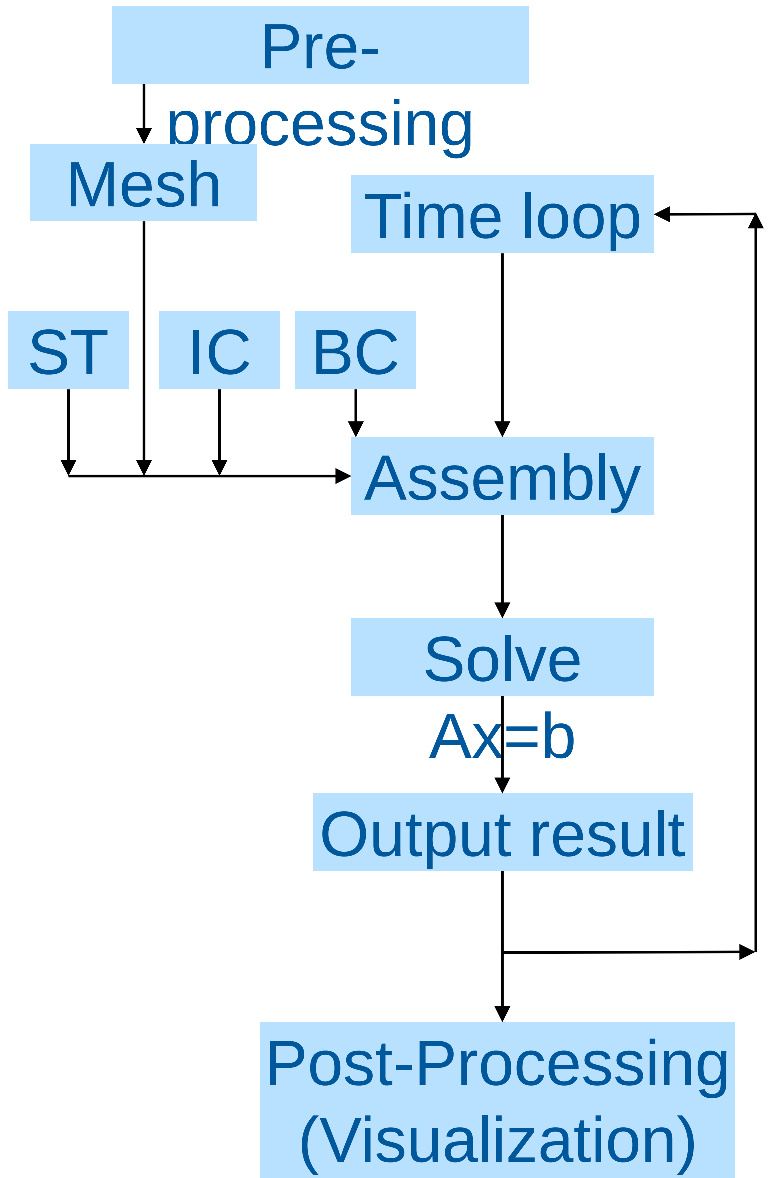
Dr. Haibing Shao
Lecture room HSZ-403, TU Dresden
Dresden, 03.06.2022

HELMHOLTZ
CENTRE FOR
ENVIRONMENTAL
RESEARCH – UFZ

Table of Content

- Finite element discretization in space
- Finite difference discretization in time
- Linear equation assembly
- Impose the Boundary condition
- Peclet and Courant number (stability)

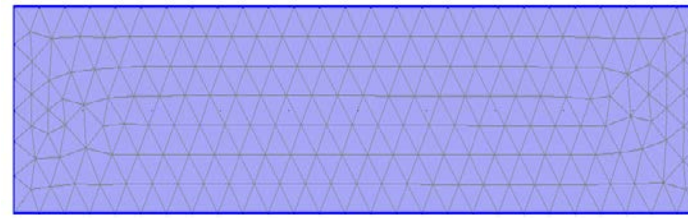
Basics of Finite Element Method



In simple words, we convert the PDE from

$$\frac{\partial c}{\partial t} + \nabla \cdot \left(-D \, \vec{\nabla} c + \vec{v} \, c \right) = s.$$

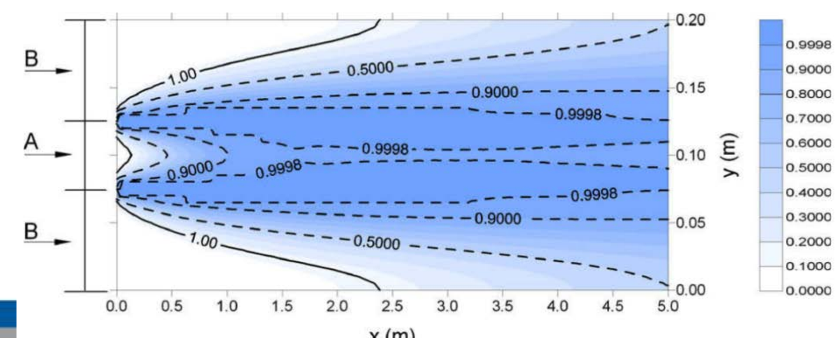
, using the topology:



, so that it is converted to linear equation Ax = b:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & . \\ . & . & . & . \quad . & . \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ . \\ b_m \end{bmatrix}$$

, and solve it, so that we get:



Finite element in space

The numerical method of FEM employs the Method of Weighted Residuals (MWR) to find the solution of a PDE.

MWR can be divided into 3 steps:

1) Approximation of the unknown function by a trial solution;
2) Definition of weighting functions;
3) Derivation of a system of algebraic equations, and solve it to find the approximation solution.

Over all connecting mesh nodes

The unknown function

The unknown on each mesh node

$$u \approx \hat{u}(x) = \sum_{i=1}^{m} \omega_i(x) C_i$$

The approximation of the unknown function

Interpolation functions (aka. Basis functions)

$$\frac{\partial}{\partial t} u + \nabla \cdot \Phi^u = q^u$$

Let Φ^u be the flux vector of conservative quantity u and q^u the source/sink term (see our first lecture about GROUNDWATER_FLOW process)

Applying the approximation, we get the weak form of the above equation as

$$\int_\Omega \omega_i \frac{\partial u}{\partial t} d\Omega + \int_\Omega \omega_i \nabla \cdot \Phi^u d\Omega = \int_\Omega \omega_i q^u d\Omega$$

In order to get rid of the special derivative of flux term, we apply the Green's theorem

$$\int_\Omega \omega_i \frac{\partial u}{\partial t} d\Omega - \int_\Omega \Phi^u \cdot \nabla \omega_i d\Omega + \oint_{\partial\Omega} \omega_i \Phi^u dS = \int_\Omega \omega_i q^u d\Omega$$

Finite element in space

$$\int_\Omega \omega_i \frac{\partial u}{\partial t} d\Omega - \int_\Omega \Phi^u \cdot \nabla \omega_i d\Omega + \oint_{\partial\Omega} \omega_i \Phi^u dS = \int_\Omega \omega_i q^u d\Omega$$

After Green's transformation, we Temporarily focusing on the space approximation, assuming time not changing.

Approximation of unknown u

$$\hat{u}(t,x) = \sum_j \omega_j(x) u_j(t)$$

Approximation of the flux of unknown u

$$\hat{\Phi}(t,x) = \sum_j \omega_j(x) \Phi_j(t)$$

So that the above equation becomes

$$\sum_i \frac{du_j}{dt} \underbrace{\int_\Omega \omega_i \omega_j d\Omega} - \sum_i \Phi^u \cdot \underbrace{\int_\Omega \omega_j \nabla \omega_i d\Omega} + \underbrace{\oint_{\partial\Omega} \omega_i \Phi^u dS} = \int_\Omega \omega_i q^u d\Omega$$

$$M_{ij} = \int_\Omega \omega_i \omega_j d\Omega \qquad K_{ij} = \int_\Omega \omega_j \nabla \omega_i d\Omega$$

This part is applied on element **Mass Matrix**

"shape-shape"

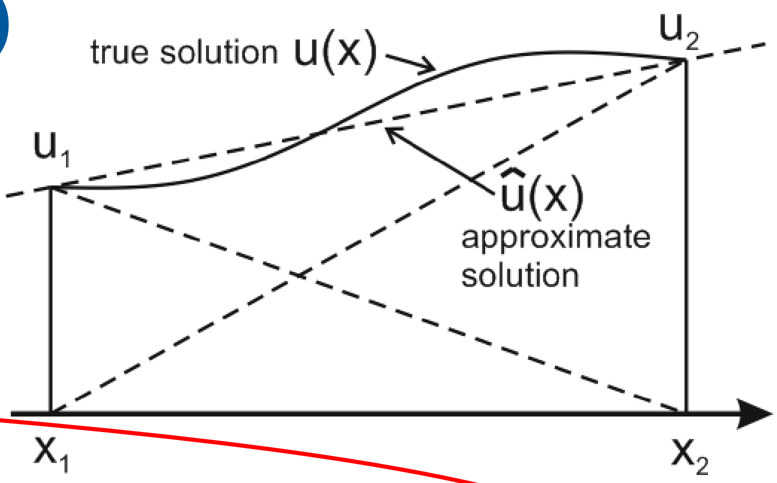
This part is applied on element **Stiffness Matrix**

"shape-dshape"

This part should zero out.
"When time is infinitely short, how much flowing-in should equal to how much flowing-out"

This part is nodal based

Shape functions (1D line element)



First let's explore how the shape function is calculated for a 1D line element.

A simple approximation of the unknown functio u(x) can be obtained by linear approximation.

$$\hat{u}(x) = a_1 + a_2 x$$

On the two ends of the line element, we assume that our unknown function produces u1 and u2 at position x1 and x2.

$$u_1 = a_1 + a_2 x_1$$
$$u_2 = a_1 + a_2 x_2$$

Write in a linear algebra form

$$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{bmatrix} 1 & x_1 \\ 2 & x_2 \end{bmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$$

Make an inversion of the expression, we get the a1 and a2 value dependency on u1 and u2.

$$\begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \frac{1}{x_2 - x_1} \begin{bmatrix} x_2 & -x_1 \\ -1 & 1 \end{bmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

Wirte back to the standard form

$$a_1 = \frac{1}{x_2 - x_1}(x_2 u_1 - x_1 u_2)$$
$$a_2 = \frac{1}{x_2 - x_1}(-u_1 + u_2)$$

Insert back here

Then we get the expression of approximated solution based on the location

$$\hat{u}(x) = \underbrace{\frac{x_2 - x}{x_2 - x_1}} u_1 + \underbrace{\frac{x - x_1}{x_2 - x_1}} u_2 = N_1(x) u_1 + N_2(x) u_2$$

N1(x) and N2(x) are the so-called shape functions.

Shape functions (1D line element, higher orders)

Linear

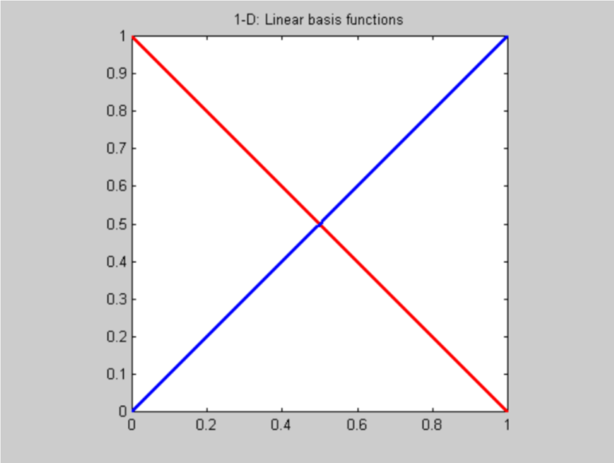
Approximation

$$\hat{u}(x) = a_1 + a_2 x$$

Shape Function

$$N_1(x) = \frac{x_2 - x}{x_2 - x_1}$$

$$N_2(x) = \frac{x - x_1}{x_2 - x_1}$$



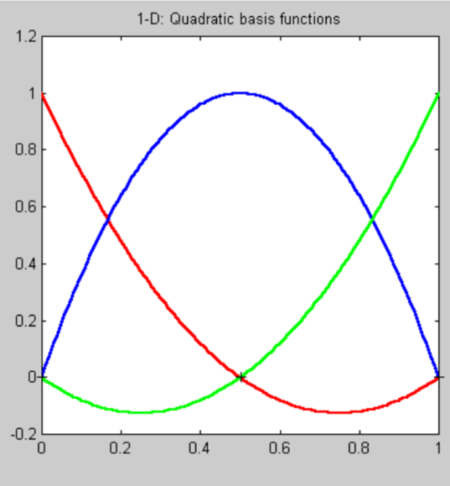
Quadradic

$$\hat{u}(x) = a_1 + a_2 x + a_3 x^2$$

$$N_1(x) = \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)}$$

$$N_2(x) = \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)}$$

$$N_3(x) = \frac{(x - x_1)(x - x_2)}{(x_3 - x_2)(x_3 - x_1)}$$



Shape functions (2D line triangle element)

Approximation: $$\hat{u}(x,y) = a_1 + a_2 x + a_3 y$$

The interpolation writes as

$$\begin{aligned} u_1 &= a_1 + a_2 x_1 + a_3 y_1 \\ u_2 &= a_1 + a_2 x_2 + a_3 y_2 \\ u_3 &= a_1 + a_2 x_3 + a_3 y_3 \end{aligned}$$

Write it in the matrix-vector form

$$\left\{ \begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array} \right\} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \left\{ \begin{array}{c} a_1 \\ a_2 \\ a_3 \end{array} \right\}$$

Inversion of the above relationship will give

$$\left\{ \begin{array}{c} a_1 \\ a_2 \\ a_3 \end{array} \right\} = \frac{1}{2A} \begin{bmatrix} x_2y_3 - x_3y_2 & x_3y_1 - x_1y_3 & x_1y_2 - x_2y_1 \\ y_2 - y_3 & y_3 - y_1 & y_1 - y_2 \\ x_3 - x_2 & x_1 - x_3 & x_2 - x_1 \end{bmatrix} \left\{ \begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array} \right\}$$

With A the surface area of the triangle

$$A = \frac{1}{2} \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$

So again our shape function act like

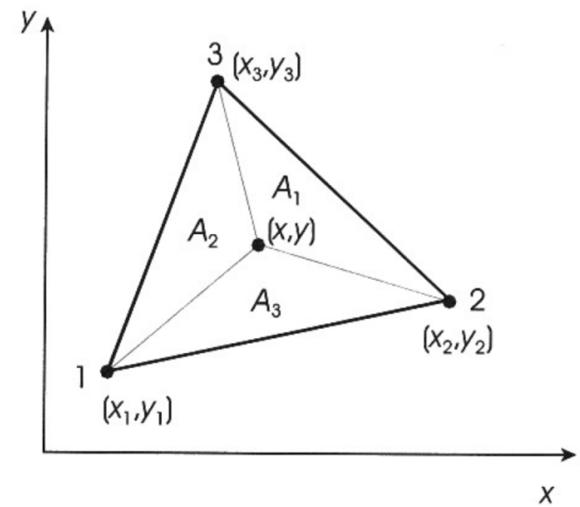
$$\hat{u}(x,y) = N_1(x,y)u_1 + N_2(x,y)u_2 + N_3(x,y)u_3$$

They can be explicitly calculated as

$$\begin{aligned} N_1(x,y) &= \frac{1}{2A}\left[(x_2y_3 - x_3y_2) + (y_2 - y_3)x + (x_3 - x_2)y\right] \\ N_2(x,y) &= \frac{1}{2A}\left[(x_3y_1 - x_1y_3) + (y_3 - y_1)x + (x_1 - x_3)y\right] \\ N_3(x,y) &= \frac{1}{2A}\left[(x_1y_2 - x_2y_1) + (y_1 - y_2)x + (x_2 - x_1)y\right] \end{aligned}$$

Or in the matrix vector form

$$\left\{ \begin{array}{c} N_1 \\ N_2 \\ N_3 \end{array} \right\} = \frac{1}{2A} \begin{bmatrix} x_2y_3 - x_3y_2 & y_2 - y_3 & x_3 - x_2 \\ x_3y_1 - x_1y_3 & y_3 - y_1 & x_1 - x_3 \\ x_1y_2 - x_2y_1 & y_1 - y_2 & x_2 - x_1 \end{bmatrix} \left\{ \begin{array}{c} 1 \\ x \\ y \end{array} \right\}$$



Finite difference in time

We start from the mass transport governing equation

$$\underbrace{\underbrace{\frac{\partial C}{\partial t}}_{\text{Change over time}}}_{\text{Time discretization}} + \underbrace{\underbrace{\nabla(-D \cdot \nabla C)}_{\text{Dispersion/Diffusion}} + \underbrace{v \cdot \nabla C}_{\text{Advection}}}_{\text{Spatial discretization}} = Q \quad \Big\}$$

Source and Rink Term, i.e. decay and reaction

- Make a difference of previous and current time step value for primary unknown.

$$C^n - \text{previous time step value}$$
$$C^{n+1} - \text{current time step value}$$

- For the time discretization part, we use forward Euler method:

$$\frac{\partial C}{\partial t} = \frac{C^{n+1} - C^n}{\Delta t}$$

- For all C values in the spatial discretization part, we apply linear interpolation btw previous and current values:

$$C = (1-\theta)C^n + \theta C^{n+1}$$

θ = 1 : C taken from the current time step, AKA implicit scheme.

θ = 0 : C taken from the previous time step, AKA explicit scheme.

Finite difference in time

We start from the mass transport governing equation

$$\underbrace{\underbrace{\frac{\partial C}{\partial t}}_{\text{Change over time}}}_{\text{Time discretization}} + \underbrace{\underbrace{\nabla(-D \cdot \nabla C)}_{\text{Dispersion/Diffusion}} + \underbrace{v \cdot \nabla C}_{\text{Advection}}}_{\text{Spatial discretization}} = Q \quad \}$$

Source and Rink Term, i.e. decay and reaction

- Make a difference of previous and current time step value for primary unknown.

C^n – previous time step value

C^{n+1} – current time step value

- For the time discretization part, we use forward Euler method:

$$\frac{\partial C}{\partial t} = \frac{C^{n+1} - C^n}{\Delta t}$$

- For all C values in the spatial discretization part, we apply linear interpolation btw previous and current values:

$$C = (1-\theta)C^n + \theta C^{n+1}$$

Handle the time derivative

$$\frac{\partial C}{\partial t} + \bigtriangledown(-D \cdot \bigtriangledown C) + v \cdot \bigtriangledown C = Q$$

$$\begin{aligned}&\frac{C^{n+1} - C^{n}}{\Delta t} + \nabla(-D \cdot \nabla((1-\theta)C^{n} + \theta C^{n+1}))\\ &+ v \cdot \nabla((1-\theta)C^{n} + \theta C^{n+1}) = Q\end{aligned}$$

We know all the previous time step value, so keep known things to the RHS and unknown things to the other.

$$\begin{aligned}&\frac{1}{\Delta t}C^{n+1} + \nabla(-D \cdot \nabla\theta C^{n+1}) + v \cdot \nabla(\theta C^{n+1})\\ &= \frac{1}{\Delta t}C^{n} - (\nabla(-D \cdot \nabla(1-\theta)C^{n}) + v \cdot \nabla((1-\theta)C^{n})) + Q\end{aligned}$$

Handle the space derivative

$$\frac{1}{\Delta t}C^{n+1} + \nabla(-D \cdot \nabla\theta C^{n+1}) + v \cdot \nabla(\theta C^{n+1})$$
$$= \frac{1}{\Delta t}C^{n} - (\nabla(-D \cdot \nabla(1-\theta)C^{n}) + v \cdot \nabla((1-\theta)C^{n})) + Q$$

Mass Term

$$\frac{1}{\Delta t}C^{n+1} \Rightarrow \int_{\Omega^e} \mathbf{N} \cdot C^{n+1} \cdot \mathbf{N} d\Omega \Rightarrow \int_{\Omega^e} \mathbf{N} \cdot \mathbf{N} d\Omega \cdot C^{n+1}$$

Dispersion/
Diffusion

$$\nabla(-D \cdot \nabla(\theta C^{n+1})) \Rightarrow \int_{\Omega^e} \nabla\mathbf{N} \cdot (-D)\nabla\mathbf{N}^{\mathbf{T}} d\Omega^e \cdot \theta C^{n+1}$$

Advection

$$v \cdot \nabla(\theta C^{n+1}) \Rightarrow \int_{\Omega^e} \mathbf{N} \cdot (\theta v C^{n+1})\nabla\mathbf{N}^{\mathbf{T}} d\Omega^e \Rightarrow \int_{\Omega^e} \mathbf{N} \cdot (v)\nabla\mathbf{N}^{\mathbf{T}} d\Omega^e \cdot \theta C^{n+1}$$

Handle the space derivative

$$\frac{1}{\Delta t}C^{n+1} + \nabla(-D \cdot \nabla\theta C^{n+1}) + v \cdot \nabla(\theta C^{n+1})$$
$$= \frac{1}{\Delta t}C^{n} - (\nabla(-D \cdot \nabla(1-\theta)C^{n}) + v \cdot \nabla((1-\theta)C^{n})) + Q$$

Mass Term

$$\frac{1}{\Delta t}C^{n+1} \Rightarrow \int_{\Omega^e} \mathbf{N} \cdot C^{n+1} \cdot \mathbf{N} d\Omega \Rightarrow \underbrace{\int_{\Omega^e} \mathbf{N} \cdot \mathbf{N} d\Omega}_{\text{Mass Matrix}} \cdot C^{n+1}$$

Dispersion/
Diffusion

$$\nabla(-D \cdot \nabla(\theta C^{n+1})) \Rightarrow \underbrace{\int_{\Omega^e} \nabla\mathbf{N} \cdot (-D)\nabla\mathbf{N}^{\mathbf{T}} d\Omega^e}_{\text{Dispersion Matrix}} \cdot \theta C^{n+1}$$

Advection

$$v \cdot \nabla(\theta C^{n+1}) \Rightarrow \int_{\Omega^e} \mathbf{N} \cdot (\theta v C^{n+1})\nabla\mathbf{N}^{\mathbf{T}} d\Omega^e \Rightarrow \underbrace{\int_{\Omega^e} \mathbf{N} \cdot (v)\nabla\mathbf{N}^{\mathbf{T}} d\Omega^e}_{\text{Advection Matrix}} \cdot \theta C^{n+1}$$

Handle the space derivative

$$\frac{1}{\Delta t}C^{n+1} + \nabla(-D\cdot\nabla\theta C^{n+1}) + v\cdot\nabla(\theta C^{n+1})$$
$$= \frac{1}{\Delta t}C^{n} - (\nabla(-D\cdot\nabla(1-\theta)C^{n}) + v\cdot\nabla((1-\theta)C^{n})) + Q$$

$$\Rightarrow \underbrace{\int_{\Omega^e}\mathbf{N}\cdot\mathbf{N}d\Omega}\cdot C^{n+1}$$

Mass Matrix

$$\Rightarrow \underbrace{\int_{\Omega^e}\nabla\mathbf{N}\cdot(-D)\nabla\mathbf{N}^{\mathbf{T}}d\Omega^e}\cdot\theta C^{n+1}$$

Dispersion Matrix

$$\Rightarrow \underbrace{\int_{\Omega^e}\mathbf{N}\cdot(v)\nabla\mathbf{N}^{\mathbf{T}}d\Omega^e}\cdot\theta C^{n+1}$$

Advection Matrix

$$\frac{1}{\Delta t}M\cdot C^{n+1} + Disp\cdot\theta C^{n+1} + Adv\cdot\theta C^{n+1}$$
$$= \frac{1}{\Delta t}M\cdot C^{n} + Disp\cdot(1-\theta)C^{n} + Adv\cdot(1-\theta)C^{n} + Q$$

$$\frac{1}{\Delta t}M\cdot C^{n+1} + (Disp + Adv)\cdot\theta C^{n+1}$$
$$= \frac{1}{\Delta t}M\cdot C^{n} + (Disp + Adv)\cdot(1-\theta)C^{n} + Q$$

K = Disp + Adv

$$(\frac{1}{\Delta t}M + \theta K)C^{n+1}$$
$$= (\frac{1}{\Delta t}M + (1-\theta)K)\cdot C^{n} + Q$$

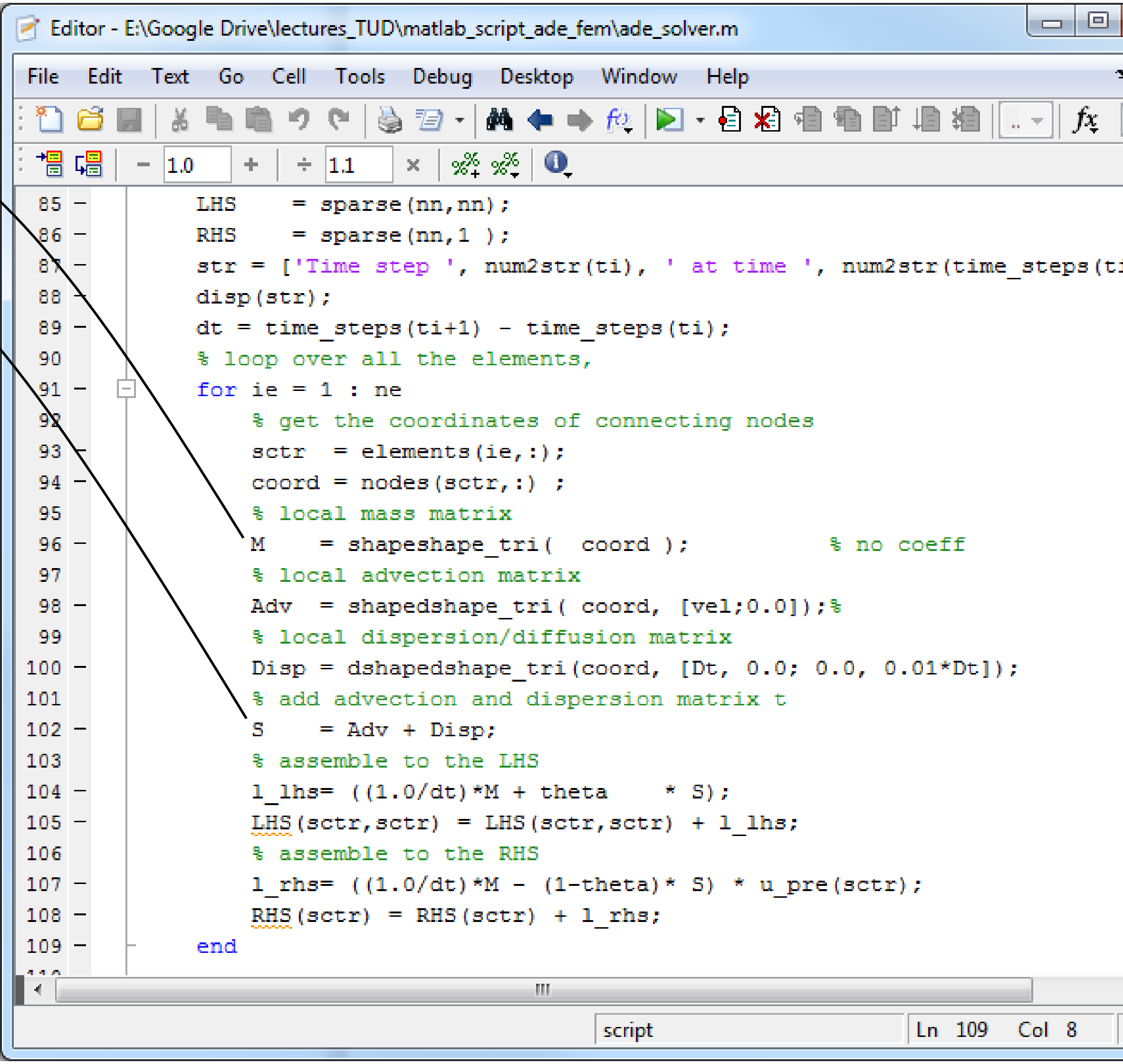
$$Ax = b$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & . \\ . & . & . & . \; . & . \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ . \\ b_m \end{bmatrix}$$

Linear equation assembly

$$\left(\frac{1}{\Delta t}M + \theta K\right)C^{n+1}$$

$$= \left(\frac{1}{\Delta t}M + (1-\theta)K\right)\cdot C^{n} + Q$$




```
    LHS     = sparse(nn,nn);
    RHS     = sparse(nn,1 );
    str = ['Time step ', num2str(ti), ' at time ', num2str(time_steps(ti
    disp(str);
    dt = time_steps(ti+1) - time_steps(ti);
    % loop over all the elements,
    for ie = 1 : ne
        % get the coordinates of connecting nodes
        sctr  = elements(ie,:);
        coord = nodes(sctr,:) ;
        % local mass matrix
        M    = shapeshape_tri(  coord );          % no coeff
        % local advection matrix
        Adv  = shapedshape_tri( coord, [vel;0.0]);%
        % local dispersion/diffusion matrix
        Disp = dshapedshape_tri(coord, [Dt, 0.0; 0.0, 0.01*Dt]);
        % add advection and dispersion matrix t
        S    = Adv + Disp;
        % assemble to the LHS
        l_lhs= ((1.0/dt)*M + theta    * S);
        LHS(sctr,sctr) = LHS(sctr,sctr) + l_lhs;
        % assemble to the RHS
        l_rhs= ((1.0/dt)*M - (1-theta)* S) * u_pre(sctr);
        RHS(sctr) = RHS(sctr) + l_rhs;
    end
```

$$Ax = b$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & . \\ . & . & . & . \; . & . \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ . \\ b_m \end{bmatrix}$$

Linear equation assembly

$$\overbrace{\left(\frac{1}{\Delta t}M + \theta K\right)}C^{n+1}$$

$$= \underbrace{\left(\frac{1}{\Delta t}M + (1-\theta)K\right)\cdot C^{n} + Q}$$

$$Ax = b$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & . \\ . & . & . & . \quad . & . \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ . \\ b_m \end{bmatrix}$$

Editor - E:\Google Drive\lectures_TUD\matlab_script_ade_fem\ade_solver.m

```
    LHS     = sparse(nn,nn);
    RHS     = sparse(nn,1 );
    str = ['Time step ', num2str(ti), ' at time ', num2str(time_steps(t
    disp(str);
    dt = time_steps(ti+1) - time_steps(ti);
    % loop over all the elements,
    for ie = 1 : ne
        % get the coordinates of connecting nodes
        sctr  = elements(ie,:);
        coord = nodes(sctr,:) ;
        % local mass matrix
        M     = shapeshape_tri(  coord );          % no coeff
        % local advection matrix
        Adv   = shapedshape_tri( coord, [vel;0.0]);%
        % local dispersion/diffusion matrix
        Disp = dshapedshape_tri(coord, [Dt, 0.0; 0.0, 0.01*Dt]);
        % add advection and dispersion matrix t
        S     = Adv + Disp;
        % assemble to the LHS
        l_lhs= ((1.0/dt)*M + theta    * S);
        LHS(sctr,sctr) = LHS(sctr,sctr) + l_lhs;
        % assemble to the RHS
        l_rhs= ((1.0/dt)*M - (1-theta)* S) * u_pre(sctr);
        RHS(sctr) = RHS(sctr) + l_rhs;
    end
```

script | Ln 109 Col 8

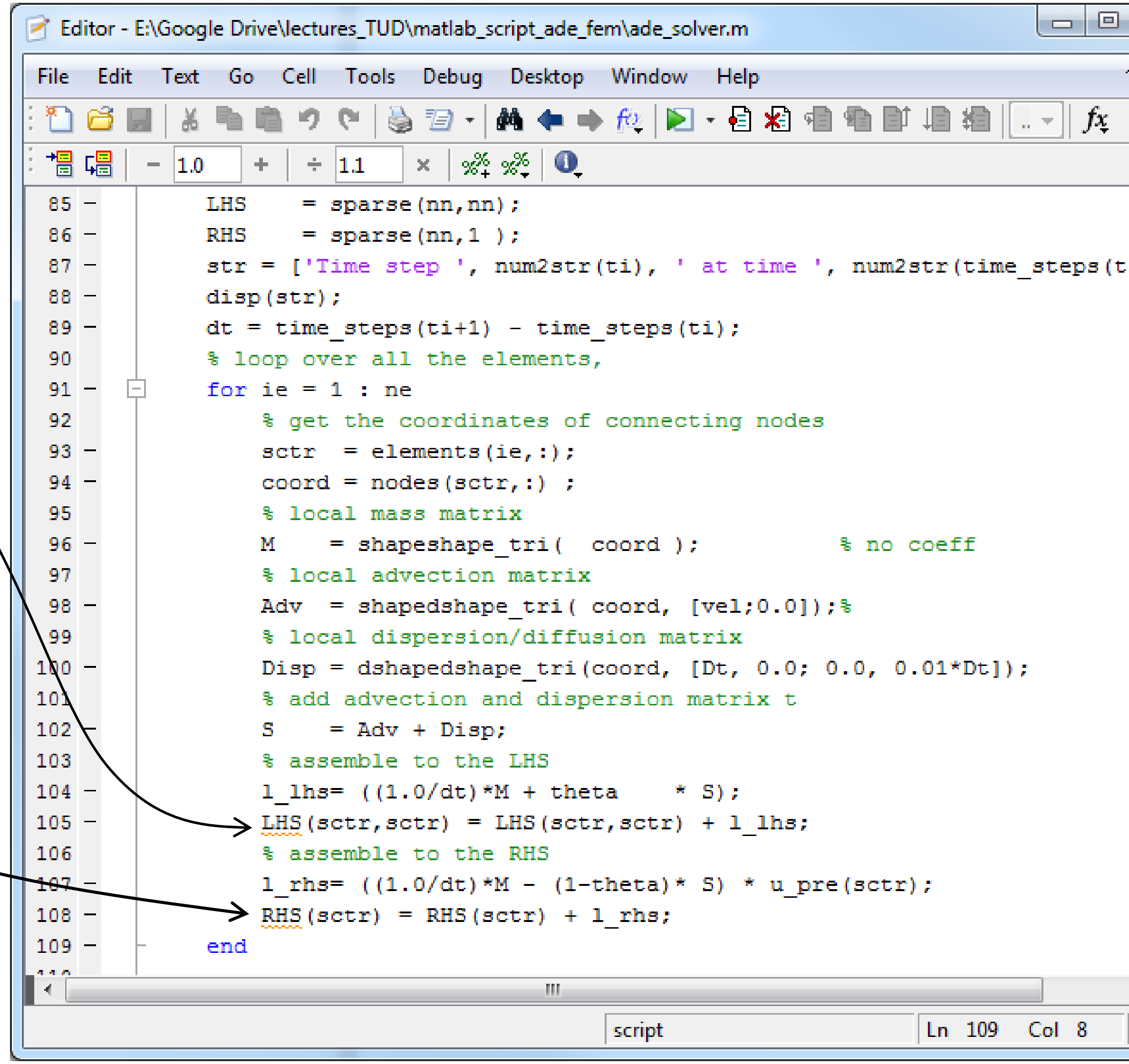
Linear equation assembly

$$\overbrace{\left(\frac{1}{\Delta t}M + \theta K\right)C^{n+1}}$$

$$= \underbrace{\left(\frac{1}{\Delta t}M + (1-\theta)K\right)\cdot C^{n} + Q}$$

$$Ax = b$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & . \\ . & . & . & . \; . & . \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ . \\ b_m \end{bmatrix}$$



Impose Boundary Condition

Assuming we know the boundary value on one of the node, how can we solve the linear equation in a way that we get the desired value on this node?

$$Ax = b$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & . \\ . & . & . & . \quad . & . \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ . \\ b_m \end{bmatrix}$$

The procedure is as follows:

1) Record the index of boundary node, say “i”.
2) RHS vector minus the multiplication of fixed boundary node value with the i-th column of LHS matrix.
3) Record the i-th row and column entry value in LHS as TMP.
4) Make i-th row of LHS all zeros.
5) Make i-th column of LHS all zeros.
6) Overwrite i-th value in RHS vector as TMP times fixed boundary value
7) Overwrite i-th row and column entry value in LHS matrix as xii.

Impose Boundary Condition

Taking the following linear equation as an example (represent a 1D Groundwater flow):

$$\begin{bmatrix} 1/2 & -1/2 & 0 & 0 & 0 \\ -1/2 & 3/2 & -1 & 0 & 0 \\ 0 & -1 & 4/3 & -1/3 & 0 \\ 0 & 0 & -1/3 & 2/3 & -1/3 \\ 0 & 0 & 0 & -1/3 & 1/3 \end{bmatrix} \begin{Bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix}$$

After imposing boundary nodes?

$$\begin{bmatrix} \_\_ & \_\_ & \_\_ & \_\_ & \_\_ \\ \_\_ & \_\_ & \_\_ & \_\_ & \_\_ \\ \_\_ & \_\_ & \_\_ & \_\_ & \_\_ \\ \_\_ & \_\_ & \_\_ & \_\_ & \_\_ \\ \_\_ & \_\_ & \_\_ & \_\_ & \_\_ \end{bmatrix} \begin{Bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{Bmatrix} = \begin{Bmatrix} \_\_ \\ \_\_ \\ \_\_ \\ \_\_ \\ \_\_ \end{Bmatrix}$$

What result do you get by solving this linear equation system?

Peclet Number

Peclet number is defined as the ratio of the rate of advection to the rate of dispersion/diffusion.

$$Pe = \frac{vL}{D}$$

- Peclet number is dimensionless.
- Peclet number reflects the ratio of advection versus diffusion. If less than one, then diffusion dominated. If more than one, then advection dominated.
- Typically, the characteristic length L refers to the length of an element.
- For the accurate solution of finite element method, the Pe number has to be kept to be less than 2.

Courant Number

AKA Courant–Friedrichs–Lewy condition

1D $$Cr = \frac{v_x \Delta t}{\Delta x} \leq Cr_{max}$$

2D $$Cr = \frac{v_x \Delta t}{\Delta x} + \frac{v_y \Delta t}{\Delta y} \leq Cr_{max}$$

- Courant number is also dimensionless
- Cr_max is typically constrained to be 2, i.e. in a given time step, one particle should not travel beyond the neighbouring element.
- Necessary condition when using explicit time integration scheme with the finite difference method.