Finite Difference Solution For Two-Dimensional Flow

Hamed Mohammadi

January 5, 2015

Contents

1 Formulation **1**

2 Finite Difference Solution For 2D Flow **4**

3 Example Flownet Using Finite Difference **10**

3.1 Solution 10

3.2 Program Code 10

3.3 Results 15

4 Flow Under a Coffer Dam **16**

4.1 Potential Lines 16

4.2 Pore-water Pressure 19

4.3 Stream Lines 19

4.4 Velocity Vectors 19

4.5 Flow Rate 19

4.6 Program Code 20

4.7 Results 28

4.7.1 Potential lines and stream lines 28

4.7.2 Velocity Vectors 28

5 Calculating Flow Using Finite Element Method **31**

5.1 About SEEP/W 31

5.2 Results 32

Preface

Flow of water through soil is one of important subjects that a civil engineer have to deal with. Seepage of water within embankment dams and under dams and coffer dams are examples of this problem.

In this writing we consider Two-Dimensional flow of water through saturated soil and solve two example problems. First problem is seepage of water under a sheet pile and the second problem is seepage of water under a coffer dam with unsymmetrical geometry and anisotropic soil. Solutions are obtained using Finite Difference Method.

Octave is used for programming the solutions. Octave is an open source and free numerical tool that has a programming language that is almost completely compatible with Matlab.

Solution to the second problem also investigated using GeoStudio SEEP/W®, which uses Finite Element Method.

Chapter 1

Formulation

The flow of water through soils is described by Laplace's equation. Flow of water through soils is analogous to steady-state heat flow and flow of current in homogeneous conductors. The popular form of Laplace's equation for two dimensional flow of water through soil is

$$k_x \frac{\partial^2 H}{\partial x^2} + k_z \frac{\partial^2 H}{\partial z^2} = 0 \tag{1.1}$$

where H is the total head and k_x and k_z are the hydraulic conductivities in the X and Z directions. Laplace's equation expresses the condition that the changes of hydraulic gradient in one direction are balanced by changes in the other directions.

The assumptions in Laplace's equation are:

- Darcy's law is valid
- Irrotational flow (vorticity) is negligible. This assumption leads to the following two-dimensional relationship in velocity gradients.

  $$\frac{\partial v_z}{\partial z} = \frac{\partial v_x}{\partial x}$$

  where v_z and v_x are the velocities in the Z and X directions respectively. This relationship is satisfied for a uniform flow field and not a general flow field.
- There is inviscid flow. This assumption means that shear stresses are neglected.

- The soil is homogeneous and saturated.
- The soil and water are incompressible (no volume change occurs).

Laplace's equation is also called the potential flow equation because the velocity head is neglected. If the soil is an isotropic material, that is, $k_x = k_z$, Laplace's equation becomes

$$\frac{\partial^2 H}{\partial x^2} + \frac{\partial^2 H}{\partial z^2} = 0 \tag{1.2}$$

The solution of any differential equation requires knowledge of the boundary conditions. The boundary conditions for most real structures are complex, so we cannot obtain an analytical solution or closed form solution for these structures. We have to resort to approximate solutions, which we can obtain using numerical methods such as finite difference, finite element, and boundary element.

The solution of Equation (1.1) depends only on the values of the total head within the flow field in the XZ plane. Let us introduce a velocity potential (ξ), which describes the variation of total head in a soil mass as

$$\xi = kH \tag{1.3}$$

where k is a generic hydraulic conductivity. The velocities of flow in the X and Z directions are

$$v_x = k_x \frac{\partial H}{\partial x} = \frac{\partial \xi}{\partial x} \tag{1.4}$$

$$v_z = k_z \frac{\partial H}{\partial z} = \frac{\partial \xi}{\partial z} \tag{1.5}$$

The inference from Equations (1.4) and (1.5) is that the velocity of flow (v) is normal to lines of constant total head, as illustrated in Figure 1.1. The direction of v is in the direction of decreasing total head. The head difference between two equipotential lines is called a potential drop or head loss.

If lines are drawn that are tangent to the velocity of flow at every point in the flow field in the XZ plane, we will get a series of lines that are normal to the equipotential lines. These tangential lines are called streamlines or flow lines (Figure 1.1). A flow line represents the path that a particle of water is expected to take in steady-state flow. A family of streamlines is represented by a stream function, $\psi\,(x,\, z)$.

The components of velocity in the X and Z directions in terms of stream function are

$$v_x = \frac{\partial \psi_s}{\partial z} \tag{1.6}$$

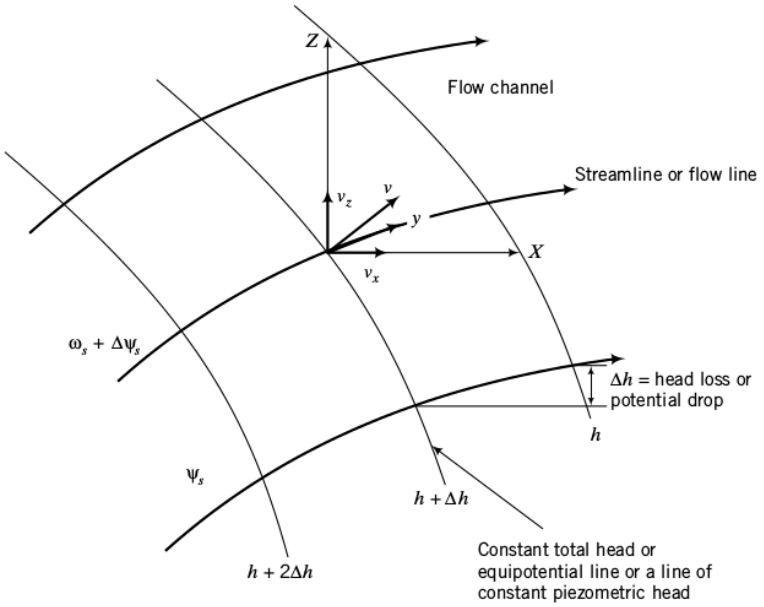


Figure 1.1: Illustration of flow terms.

$$v_z = \frac{\partial \psi_s}{\partial x} \tag{1.7}$$

Since flow lines are normal to equipotential lines, there can be no flow across flow lines. The rate of flow between any two flow lines is constant. The area between two flow lines is called a flow channel (Figure 1.1). Therefore, the rate of flow is constant in a flow channel.

Chapter 2

Finite Difference Solution For 2D Flow

We solve Laplace's equation to determine two-dimensional confined flow through soils. Let us consider a grid of a flow domain, as shown in Figure 2.1, where (i, j) is a nodal point.

Using Taylor's theorem, we have

$$k_x \frac{\partial^2 H}{\partial x^2} + k_z \frac{\partial^2 H}{\partial z^2} = \frac{k_x}{\Delta x}\left(h_{i+1,j} + h_{i-1,j} - 2h_{i,j}\right) + \frac{k_z}{\Delta z^2}\left(h_{i,j+1} + h_{i,j-1} - 2h_{i,j}\right) = 0 \tag{2.1}$$

Let $\alpha = k_x/k_z$ and $\Delta x = \Delta z$ (i.e., we subdivide the flow domain into a square grid). Then solving for $h_{i,j}$ from Equation (2.1) gives

$$h_{i,j} = \frac{1}{2\left(1+\alpha\right)}\left(\alpha h_{i+1,j} + \alpha h_{i-1,j} + h_{i,j-1} + h_{i,j-1}\right) \tag{2.2}$$

For isotropic conditions, $\alpha = 1\,(k_x = k_z)$ and Equation (2.2) becomes

$$h_{i,j} = \frac{1}{4}\left(h_{i+1,j} + h_{i-1,j} + h_{i-1,j} + h_{i,j-1}\right) \tag{2.3}$$

Since we are considering confined flow, one or more of the boundaries would be impermeable.

Flow cannot cross impermeable boundaries and, therefore, for a horizontal impermeable surface,

$$\frac{\partial h}{\partial x} = 0 \tag{2.4}$$

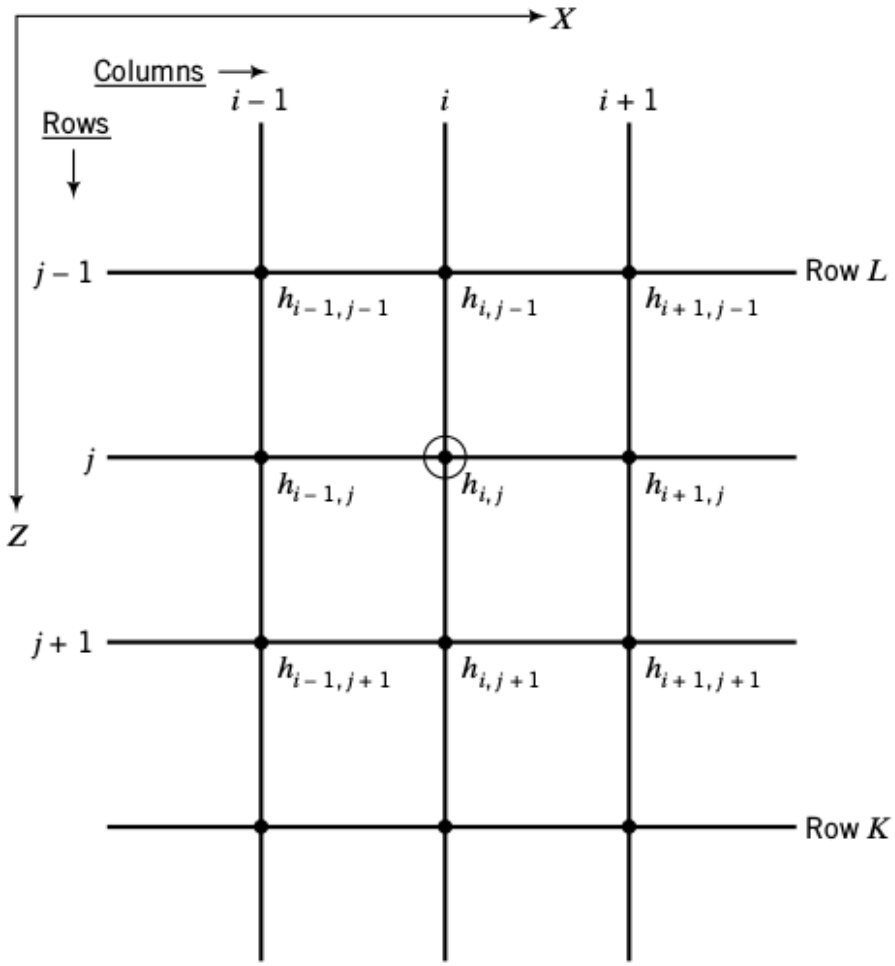


Figure 2.1: A partial grid of the flow domain

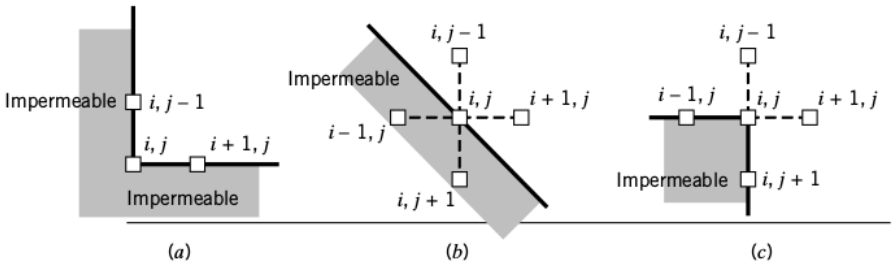


Figure 2.2: Three types of boundary encountered in practice

The finite difference form of Equation (2.4) is

$$\frac{\partial h}{\partial x} = \frac{1}{2\Delta x}\left(h_{i,j+1} - h_{i,j-1}\right) = 0 \tag{2.5}$$

Therefore, $h_{i,j+1} = h_{i,j-1}$ and, by substituting in Equation (2.3), we get

$$h_{i,j} = \frac{1}{4}\left(h_{i+1,j} + h_{i-1,j} + 2h_{i,j-1}\right) \tag{2.6}$$

Various types of geometry of impermeable boundaries are encountered in practice, three of which are shown in Figure 2.2. For Figure 2.2a, b, the finite difference equation is

$$h_{i,j} = \frac{1}{2}\left(h_{i+1,j} + h_{i,j-1}\right) \tag{2.7}$$

and, for Figure 2.2c,

$$h_{i,j} = \frac{1}{3}\left(h_{i,j-1} + h_{i+1,j} + h_{i,j+1} + \frac{1}{2}h_{i-1,j} + \frac{1}{2}h_{i,j+1}\right) \tag{2.8}$$

The pore-water pressure at any node ($u_{i,j}$) is

$$u_{i,j} = \gamma_w\left(h_{i,j} - z_{i,j}\right) \tag{2.9}$$

where $z_{i,j}$ is the elevation head.

Contours of potential heads can be drawn from discrete values of $h_{i,j}$. The finite difference equations for flow lines are analogous to the potential lines; that is, ψ_s replaces h in the above equations and the boundary conditions are specified for ψ_s rather than for h.

The horizontal velocity of flow at any node ($v_{i,j}$) is given by Darcy's law:

$$v_{i,j} = k_x i_{i,j}$$

where $i_{i,j}$ is the hydraulic gradient expressed as

$$i_{i,j} = \frac{(h_{i+1,j} - h_{i-1,j})}{2\Delta x} \tag{2.10}$$

Therefor,

$$v_{i,j} = \frac{k_x}{2\Delta x}(h_{i+1,j} - h_{i-1,j}) \tag{2.11}$$

The flow rate, q, is obtained by considering a vertical plane across the flow domain. Let L be the top row and K be the bottom row of a vertical plane defined by column i (Figure 2.1). Then the expression for q is

$$q = \frac{k_x}{4}\left(h_{i+1,L} - h_{i-1,L} + 2\sum_{j=L+1}^{K-1}(h_{i+1,j} - h_{i-1}) + h_{i+1,K} - h_{i-1,K}\right) \tag{2.12}$$

The procedure to determine the distribution of potential head, flow, and pore-water pressure using the finite difference method is as follows:

1. Divide the flow domain into a square grid. Remember that finer grid gives more accurate solutions than coarser grids, but are more tedious to construct and require more computational time. If the problem is symmetrical, you only need to consider one-half of the flow domain. For example, the sheet pile wall shown in Figure 2.3 is symmetrical about the wall and only the left half may be considered. The total flow domain should have a width of at least four times the thickness of the soil layer. For example, if D is the thickness of the soil layer (Figure 2.3), then the minimum width of the left half of the flow domain is $2D$.

2. Identify boundary conditions, for example impermeable boundaries (flow lines) and permeable boundaries (equipotential lines).

3. Determine the heads at the permeable or equipotential boundaries. For example, the head along the equipotential boundary AB (Figure 2.3) is ΔH. Because of symmetry, the head along nodes directly under the sheet pile wall (EF) is $\Delta H/2$.

4. Apply the known heads to corresponding nodes and assume reasonable

initial values for the interior nodes. You can use linear interpolation for potential heads of the interior nodes.

5. Apply Equation (2.2) (or Equation (2.3) if soil is isotropic), to each node except (a) at impermeable boundaries, where you should use Equation (2.6), (b) at corners, where you should use Equations (2.7) and (2.8) for the corners shown in Figure (2.2)a-c, and (c) at nodes where the heads are known.

6. Repeat item 5 until the new value at a node differs from the old value by a small numerical tolerance, for example, $0.001\,m$.

7. Arbitrary select a sequential set of nodes along a column of nodes and calculate the flow, q, using Equation (2.12). It is best to calculate $q' = q$ for a unit permeability value to avoid too many decimal points in the calculations.

8. Repeat items 1 to 6 to find the flow distribution by replacing heads by flow q'. For example, the flow rate calculated in item 7 is applied to all nodes along AC and CF (Figure (2.3)). The flow rate at nodes along BE is zero.

9. Calculate the pore-water pressure distribution by using Equation (2.9).

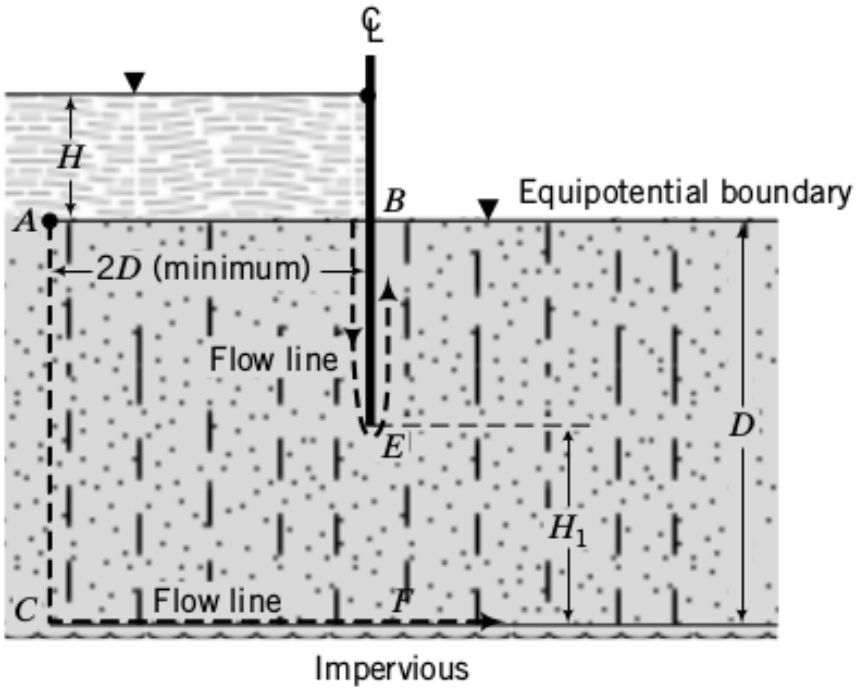


Figure 2.3: A sheet pile wall

Chapter 3

Example Flownet Using Finite Difference

In this example we want to determine the flow rate under the sheet pile (Figure (3.1)) and the pore-water pressure distribution using the finite difference method.

3.1 Solution

We first divide the flow domain into grid as shown in Figure 3.2.

Since the problem is symmetrical, we perform calculations only for one-half of domain. We use the left half of the flow domain of width $2D = 2 \times 12 = 24\,m$. Use a grid $2 \times 2\,m$. The grid is shown as $ABCDE$ in Figure 3.2.

Permeable boundaries AB and CD are equipotential lines. Impermeable boundaries BC, AE and DE are flow lines.

Along AB, the head difference is $4-1 = 3\,m$. Along CD, the head difference is $3/2 = 1.5\,m$.

In all other grid points we assume a reseanable value. We can use linear interpolation. Next we use five point finite difference formula, and using Gauss-Jordan iterative method the solution will be determined.

3.2 Program Code

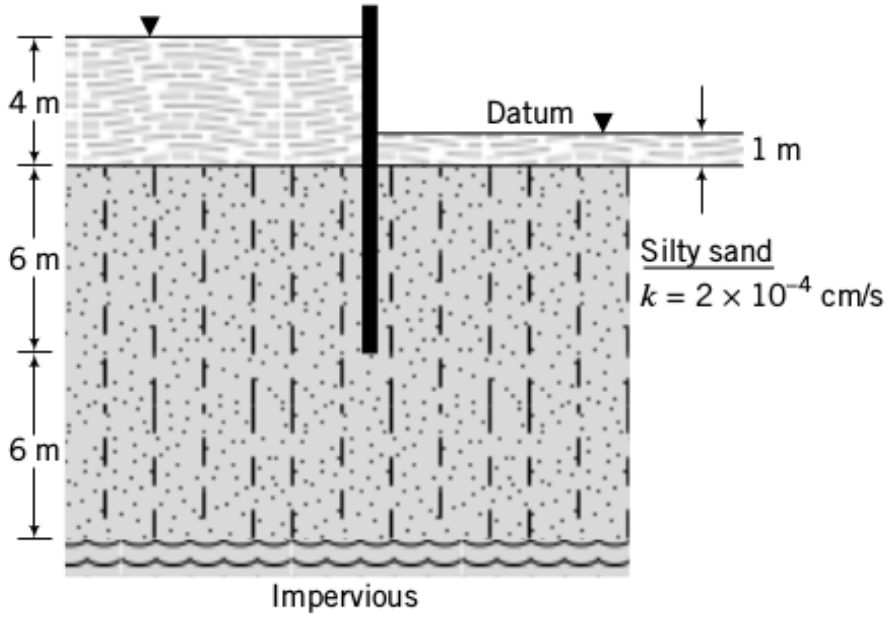


Figure 3.1: Flow under a sheet pile

```
,
#!/usr/bin/octave -qf

AB = [3.00 3.00 3.00 3.00 3.00 3.00 3.00 3.00 3.00 3.00 3.00 3.00
    3.00];
ED = [2.52 2.44 2.36 2.28 2.20 2.12 2.04 1.96 1.88 1.80 1.72 1.64
    1.5];
AE = [3.00; 2.92; 2.84; 2.76; 2.68; 2.60; 2.52];
BD = [3.00; 2.50; 2.00; 1.50; 1.50; 1.50; 1.50];

h = ones(7,13);
h(1, :) = AB;
h(:, 13) = BD;
h(:, 1) = AE;
h(7, :) = ED

for k = 2:12
    d = (h(1,k)-h(7,k))/6.0;
    for l = 2:6
        h(l,k) = h(l-1, k) - d;
    end
```

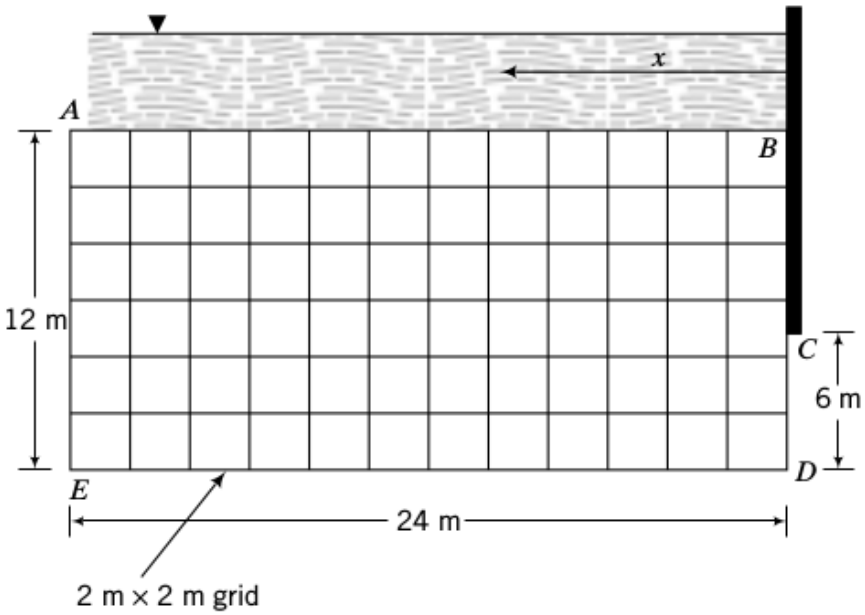


Figure 3.2: Grid of flow domain for sheet pile

```
end

max_it = 1000;
tol = 0.00000001;
toln = tol;

%iteration
rel_err = 1;
itnum = 0;

while ((rel_err>toln) & (itnum<=max_it))
    err = 0;
    umax = 0;
    for j = 2 : 12
        for i = 2 : 6
            temp = (h(i+1,j)+h(i-1,j)+h(i,j+1)+h(i, j-1))/4;
            dif = abs(temp - h(i, j));

            if (err <= dif)
                err = dif;
            end
```

```
            h(i, j) = temp;
        end
    end
    itnum = itnum +1;
    rel_err = err;

    for i = 2 :6
        h( i, 1 ) = 0.25 *( h( i -1, 1)+h(i+1, 1) +2*h(i, 2));
    end

    h(7, 1) = 0.5 *(h(6, 1)+h(7,2));

    for j = 2 : 12
        h( 7, j) = 0.25 * (h(7, j-1)+h(7,j+1)+2*h(6, j));
    end

    h(2, 13) = 0.25 * ( h(1, 13)+ h(3, 13) + 2 * h(2, 12));
    h(3, 13) = 0.25 * ( h(2,13) + h(4, 13) + 2 * h(3, 12));

end

z = ones(7,13)*(-1);
for i = 2: 7
    z(i, : ) =  z(i-1, 1) - 2 ;
end

u = 9.806 * (h-z);

q1 = 0;
for i = 2 : 12
    q1 = q1 + ( h(2, i) - h(4,i));
end
q1
q_per_unit_k = 0.25 * ((h(2, 1)-h(4,1))+2*q1+(h(2,13)-h(4,13)))

S_AE = [1.63; 1.63; 1.63; 1.63; 1.63; 1.63; 1.63];
S_BD = [0.00; 0.00; 0.00; 0.00; 0.55; 1.09; 1.63];

s = zeros( 7, 13);
s(:,1) = S_AE;
s(:,13) = S_BD;

for i = 1 : 7
    d = (s(i, 1) - s(i, 13))/12;
    for j = 2 : 12
```

```
        s(i, j) = s(i, j-1) - d;
    end
end

max_it = 1000;
tol = 0.00000001;
toln = tol;

%iteration
rel_err = 1;
itnum = 0;

while ((rel_err>toln) & (itnum<=max_it))
    err = 0;
    umax = 0;
    for j = 2 : 12
        for i = 2 : 6
            temp = (s(i+1,j)+s(i-1,j)+ s(i,j+1)+ s(i, j-1))/4;
            dif = abs(temp - s(i, j));

            if (err <= dif)
                err = dif;
            end

            s(i, j) = temp;
        end
    end
    itnum = itnum +1;
    rel_err = err;

    s(5, 13) = 0.25 * ( s(4, 13)+ s(6, 13) + 2 * s(5, 12));
    s(6, 13) = 0.25 * ( s(5,13) + s(7, 13) + 2 * s(6, 12));

    for j = 2 : 12
        s( 1, j) = 0.25 * (s(1, j-1)+s(1,j+1)+2*s(2, j));
    end
end


h
u
s

x = [-24:2:0];
y = [0:-2:-12];
```

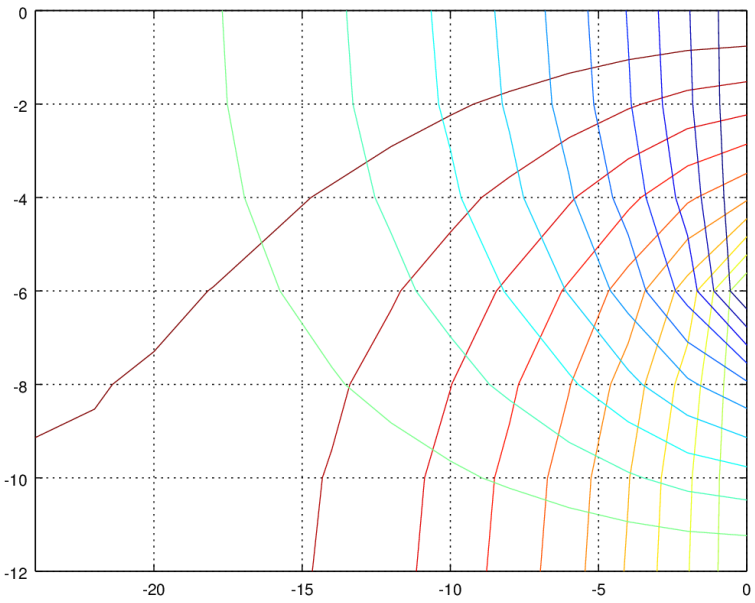


Figure 3.3: Potential lines and stream lines

```
hold on;
grid on;
contour(x, y, h);
contour(x, y, s);
print -dpng plot.png
```

3.3 Results

After executing the program we obtain plot of potential lines and stream line as shown in Figure 3.3. Discharge under the sheet pile is calculated to be $Q = 3.2 \times 10^{-6}\, m^2/s$

Chapter 4

Flow Under a Coffer Dam

In this chapter we solve the problem of seepage under a coffer dam in which the soil mass is not isotropic. we have $k_z = k_x/2$. The geometry of the problem is also not symmetric as shown in Figure 4.1. Thus we can't do calculations only for the half of the soil domain. also since impermeable base is not horizontal we should use especial grids to overcome this situation. Next we formulate the problem using five point finite difference formulla to get a working formulla for this especial type of problem.

4.1 Potential Lines

As seen before flow through soil obeys Lapace's equation.

$$k_x \frac{\partial^2 H}{\partial x^2} + k_z \frac{\partial^2 H}{\partial z^2} = 0 \tag{4.1}$$

Using Taylor's theorem, we have

$$\frac{k_x}{\Delta x^2}\left(h_{i+1,j} + h_{i-1,j} - 2h_{i,j}\right) + \frac{k_z}{\Delta z^2}\left(h_{i,j+1} + h_{i,j-1} + 2h_{i,j}\right) = 0 \tag{4.2}$$

In this problem we have $k_z = k_x/2$. Substituting for k_z in above equation we can obtain

$$\frac{k_x}{\Delta x^2}\left(h_{i+1,j} + h_{i-1,j} - 2h_{i,j}\right) + \frac{k_x}{2\Delta z^2}\left(h_{i,j+1} + h_{i,j-1} + 2h_{i,j}\right) = 0 \tag{4.3}$$

For special case of geometry of this problem its more convenient to set $\Delta x =$

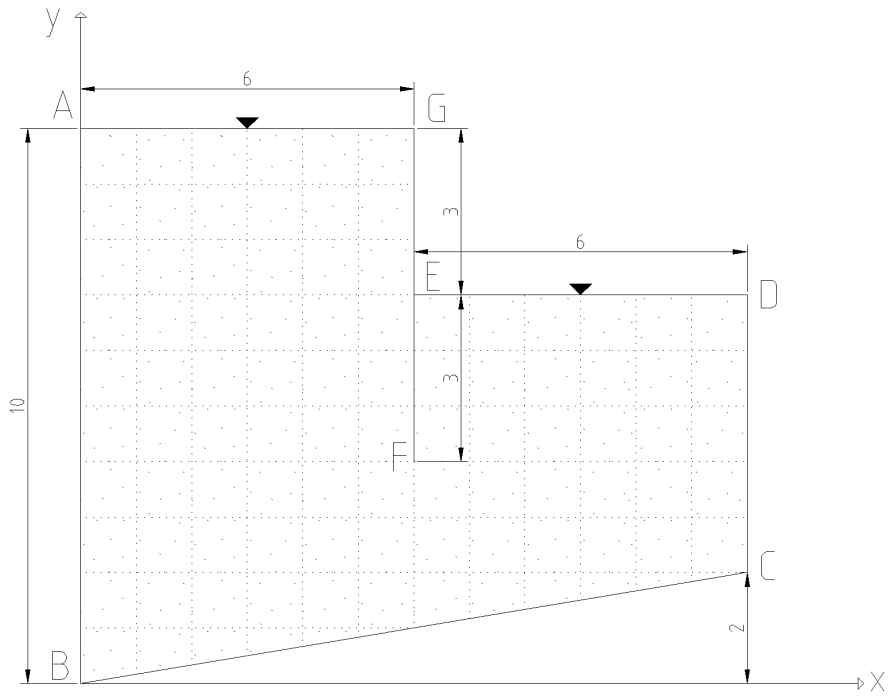


Figure 4.1: Seepage under a coffer dam with anisotropic soil.

$6\Delta z$, therefore

$$\Delta x^2 = 36\Delta z^2 \rightarrow \Delta z^2 = \frac{1}{36}\Delta x \tag{4.4}$$

Substituting back for Δz in potential equation gives

$$\frac{k_x}{\Delta x^2}\left(h_{i+1,j} + h_{i-1,j} - 2h_{i,j}\right) + \frac{18k_z}{\Delta x^2}\left(h_{i,j+1} + h_{i,j-1} + 2h_{i,j}\right) = 0 \tag{4.5}$$

and

$$\frac{k_x}{\Delta x^2}\left(h_{i+1,j} + h_{i-1,j} - 2h_{i,j} + 18h_{i,j+1} + 18h_{i,j-1} + 36h_{i,j}\right) = 0 \tag{4.6}$$

k_x and Δx are not equal to zero, therefore the term in prenteses is equal to zero. Solving for $h_{i,j}$ (see Figure 2.1), we can obtain

$$h_{i,j} = \frac{1}{38}\left(h_{i+1,j} + h_{i-1,j} + 18h_{i,j+1} + 18h_{i,j-1}\right) \tag{4.7}$$

As mentioned earlier for impermeable boundaries we have $h_{i,j+1} = h_{i,j-1}$, and $h_{i+1,j} = h_{i-1,j}$. In present problem we face five types of boundary conditions.

1. Constant head upstream and downstream soil surface. These surfaces are potential line and stream lines connects them with right angle. Potential head at upstream is 4 and at downstream is zero.

2. Vertical impermeable boundary conditions at left and right and at sheet pile. For these boundaries using Equation (4.7) we have

$$h_{i,j} = \frac{1}{38}\left(2h_{i+1,j} + 18h_{i,j+1} + 18h_{i,j-1}\right) \tag{4.8}$$

3. Left corner

$$h_L = \frac{36}{38}h_{i,j-1} \tag{4.9}$$

in which h_L refers potential head at left corner.

4. Right corner

$$h_R = \frac{1}{19}\left(h_{i-1,j} + 18h_{i,j-1}\right) \tag{4.10}$$

5. Inclined impermeable subsurface. For this case finite difference formula becomes

$$h_{i,j} = \frac{1}{19}\left(h_{i-1,j} + 18h_{i,j-1}\right) \tag{4.11}$$

We can plot contour of potential heads from discrete values of $h_{i,j}$.

4.2 Pore-water Pressure

The pore-water pressure at any node $(u_{i,j})$ is

$$u_{i,j} = \gamma_w \left(h_{i,j} - z_{i,j}\right) \tag{4.12}$$

where $z_{i,j}$ is the elevation head.

4.3 Stream Lines

The finite difference equation for flow lines are analogous to the potential lines: that is, ψ_s replaces in the above equations and the boundary conditions are specified for ψ_s, rather than h.

4.4 Velocity Vectors

The horizontal velocity of flow at any node $(v_{i,j})$ is given by Darcy's law:

$$v_{i,j} = k_x i_{i,j} \tag{4.13}$$

where $i_{i,j}$ is the hydraulic gradient expressed as

$$i_{i,j} = \frac{(h_{i+1,j} - h_{i-1,j})}{2\Delta x} \tag{4.14}$$

Therefore,

$$v_{i,j} = \frac{k_x}{2\Delta x} \left(h_{i+1,j} - h_{i-1,j}\right) \tag{4.15}$$

Similarly for vertical component of flow velocity as each node we can obtain

$$w_{i,j} = \frac{k_z}{2\Delta z} \left(h_{i,j+1} - h_{i,j-1}\right) \tag{4.16}$$

4.5 Flow Rate

The flow rate, q, is obtained by considering a vertical plane across the flow domain. Let L be the top row and K be the bottom row of a vertical plane

defined by column i. Then the expression

4.6 Program Code

A program for solving this problem using finite difference method is written using Octave. Octave is a numerical programming language that is very similar to Matlab.

```
   ,
#!/usr/bin/octave -qf

kx = 10e-5;
ky = 0.5*10e-5;

n_j = 10/(1/6)+1;
n_i = 14;

h = zeros(n_j,n_i);

for i = 1:7
    h(1,i) = 3;
end

dh = 3/102;
for j = 2:61
    h(j,1) = h(j-1,1) - dh;
end

for j = 20:49
    h(j,14) = h(j-1,14) + dh;
end

k = 1;
for i = 2:13
    if (i == 8)
        h(61-k+1,i) = h(61-k+1,i-1);
    else
        h(61-k,i) = h(61-k+1,i-1) - dh;
        k++;
    end
end

for j = 37:54
    h(j,7)=h(j,8)=4/3;
end
```

```
dh = (h(1,7)-h(37,7))/36;
for j = 2:36
    h(j,7) = h(j-1,7) - dh;
end

dh = (h(37,7)-h(19,8))/18;
for j = 20:36
    h(j,8) = h(j-1,8)+dh;
end

for j = 2:55
    dh = (h(j,1)-h(j,7))/6;
    for i = 2:6
        h(j,i) = h(j,i-1) - dh;
    end
end

for j = 20:49
    dh = (h(j,14)-h(j,8))/6;
    for i = 9:13
        h(j,i) = h(j,i-1)+dh;
    end
end

k=2;
for j = 50:53
    l = 14-k;
    dh = (h(j,l)-h(j,8))/(6-k+1);
    for i = 9:(14-k)
        h(j,i) = h(j,i-1)+dh;
    end
    k++;
end

k=2;
for j = 56:59
    l=7-k;
    dh = (h(j,1)-h(j,l))/(6-k+1);
    for i = 2:(7-k)
        h(j,i)=h(j,i-1)-dh;
    end
    k++;
end

h;
```

```
max_it = 10000;
toln = 0.00001;
h_max = 3;
rel_err = 1;
it_num = 0;

while( (rel_err>toln) && (it_num <=max_it))
    err = 0;
    for i= 2: 6
        for j=2:55
            temp = (1/38)*(18*h(j+1, i)+18*h(j-1,i)+h(j,i+1)+h(j,i
                -1));
            dif = abs(temp-h(j,i));
            if(err<=dif)
                err = dif;
            end
            h(j,i) = temp;
        end
    end

    err = 0;
    for i= 9: 13
                for j=20:49
                        temp = (18*h(j+1, i)+18*h(j-1,i)+h(j,i+1)+h
                            (j,i-1))/38;
                        dif = abs(temp-h(j,i));
                        if(err<=dif)
                                err = dif;
                        end
                        h(j,i) = temp;
                end
        end

    k=0;
    for j = 56:59
        for i = 2:(5-k)
            h(j,i) = (1/38)*(h(j,i-1)+h(j,i+1)+18*h(j-1,i)+18*h(j
                +1,i));
        end
        k++;
    end

    k=0;
    for j = 50:53
        for i = 9:(12-k)
```

```
            h(j,i) = (1/38)*(h(j,i-1)+h(j,i+1)+18*h(j-1,i)+18*h(j
                +1,i));
        end
        k++;
    end


    h(61,1) = h(60,1);
    h(49,14) = (1/19) * (h(49,13)+18*h(48,14));

    for j = 2:60
        h(j,1) = (1/38)*(18*h(j-1,1)+18*h(j+1,1)+2*h(j,2));
    end

    k=1;
    for i = 2:13
        if ( i == 7)
            h(55,i) = (1/19)*(h(55, i-1)+18*h(54, i));
            k++;
            continue;
        end

        if ( i == 8 )
            h(55,i) = h(55, i-1);
            continue;
        end

        h(61-k,i) = (1/19)*(h(61-k, i-1) +18*h(61-k-1, i));
        k++;
    end

    for j = 2:36
        h(j,7)= (1/38)*(18*h(j-1,7)+18*h(j+1,7)+2*h(j,6));
    end

    for j = 32:36
        h(j, 8)=(1/38)*(18*h(j-1,8)+18*h(j+1,8)+2**h(j,9));
    end

    for j = 20:48
        h(j,14) = (1/38)*(18*h(j-1,14)+18*h(j+1,14)+2*h(j,13));
    end

    h(37,7) = h(37,8) = (1/38) * (h(37,6)+h(37,9)+2*18*h(38,7));

    for j = 38:54
```

```
        h(j,7)=h(j,8)=(1/38)*(h(j,6)+h(j,9)+18*h(j-1,7)+18*h(j+1,7)
            );
    end

    rel_err = err/h_max;
    it_num++;

end

z = zeros(61,14);

for j = 2:61
    z(j,:) = z(i-1,1)-(1/6);
end

z = z.*h./h;

u = 9806*(h-z);

unit_q = 0;
q1 = 0;

for i = 2:6
    q1=q1+(h(30,i)-h(32,i));
end

unit_q = (1/12)*((h(30,1)-h(32,1))+2*q1+(h(30,7)-h(32,7)))
q = unit_q * ky

s=zeros(61,14);

for j=1:61
    s(j,1) = unit_q;
end

k=1;
for i=2:7
    s(61-k,i)=unit_q;
    k++;
end
s(55,8)=s(55,7);

k=1;
for i = 9:13
    s(55-k,i)=unit_q;
    k++;
```

```
end

for j = 19:49
    s(j, 14) = unit_q;
end

for j = 1:37
    s(j,7) = 0.0;
end

for j = 19:37
    s(j,8)= 0.0;
end

ds = (s(55,7)-s(37,7))/(55-37)
for j= 38:54
    s(j,7)=s(j,8)=s(j-1,7)+ds;
end

for j = 1:54
    dsx = (s(j,1)-s(j,7))/6.0;
    for i = 2 : 6
        s(j,i) = s(j,i-1) -dsx;
    end
end

for j = 19:48
    dsx=(s(j,14)-s(j,8))/6.0;
    for i = 9:13
        s(j,i)=s(j,i-1)+dsx;
    end
end

s_max = unit_q;
rel_err = 1;
it_nums = 0;

while( (rel_err>toln) && (it_nums <=max_it))
    err = 0;
    for i= 2: 6
        for j=2:55
            temp = (1/4)*(s(j+1, i)+s(j-1,i)+s(j,i+1)+s(j,i-1));
            dif = abs(temp-s(j,i));
            if(err<=dif)
                err = dif;
            end
```

```
            s(j,i) = temp;
        end
    end

    err = 0;
    for i= 9: 13
                for j=19:49
                        temp = (1/4)*(s(j+1, i)+s(j-1,i)+s(j,i+1)+s
                            (j,i-1));
                        dif = abs(temp-s(j,i));
                        if(err<=dif)
                                err = dif;
                        end
                        s(j,i) = temp;
                end
        end

    k=1;
    for j=56:60
        for i=2:(6-k)
            s(j,i)=(1/4)*(s(j-1,i)+s(j+1,i)+s(j,i-1)+s(j,i+1));
        end
        k++;
    end

    k=1;
    for j=50:54
        for i=9:(13-k)
            s(j,i)=(1/4)*(s(j-1,i)+s(j+1,i)+s(j,i-1)+s(j,i+1));
        end
        k++;
    end

    for i = 2:6
        s(1,i) = (1/4)*(s(1,i-1)+s(1,i+1)+2*s(2,i));
    end

    for i = 9 : 13
        s(19,i) = (1/4)*(s(19,i-1)+s(19,i+1)+2*s(20,i));
    end

    for j =38:54
        s(j,7)=s(j,8)=(1/4)*(s(j-1,7)+s(j+1,7)+s(j,6)+s(j,9));
    end

    rel_err = err/s_max;
```

```
    it_nums++;
end

it_num
s

hold on;
grid minor;

x = [[-6:1:-1] -eps eps [1:1:6]];
y = [0:-1/6:-10];

[cs hs] = contour(x,y,h,15);
clabel(cs, hs);

[cs hs] = contour(x,y,s,15);
title("Seapage␣a␣coffer␣dam.␣Potential␣and␣streamlines");
xlabel("x␣(m)");
ylabel("y␣(m)");
print -dpng seapage.png

hold off;
clf;

v=zeros(11,14);
u=zeros(11,14);

%for j=2:9
%    l=j*6+1;
%    u(j,1)=(ky*(h(l+1,1)+h(l-1,1)))/2;
%end

%for j=2:7
%    l=j*6+1;
%    u(j,1)=(ky*(h(l+1,7)-h(l-1,1)))/2;
%end

%for j=4:7
%    l=j*6+1;
%    u(j,8)=(ky*(h(l+1,8)-h(l-1,8)))/2;
%end

%for j=4:9
%    l=j*6+1;
%    u(j,14)=(ky*(h(l+1,14)-h(l-1,14)))/2;
%end
```

```
for j =1:9
    for i=2:6
        l = j*6+1;
        v(j,i)=kx*(h(l,i-1)-h(l,i+1))/2;
        u(j,i)=ky*(h(l+1,i)-h(l-1,i))/2;
    end
end

for j=4:8
    for i=9:13
        l=j*6+1;
        v(j,i)=kx*(h(l,i-1)-h(l,i+1))/2;
        u(j,i)=ky*(h(l+1,i)-h(l-1,i))/2;
    end
end


xv =[[-6:1:-1] -eps eps [1:1:6]];
yv =[0:-1:-10];
quiver(xv,yv,v,u,2);

print -dpng vector.png

[c h] = contour(x,y,u);

clabel(c, h);

print -dpng pore_pressure.png
```

4.7 Results

After running the program code the following results was obtained. Discharge is computed to be $Q = 3.75 \times 10^{-6}\, m^3/s$.

4.7.1 Potential lines and stream lines

Potential lines and stream lines are shown in Figure 4.2

4.7.2 Velocity Vectors

Velocity vectors are shown in Figure 4.1

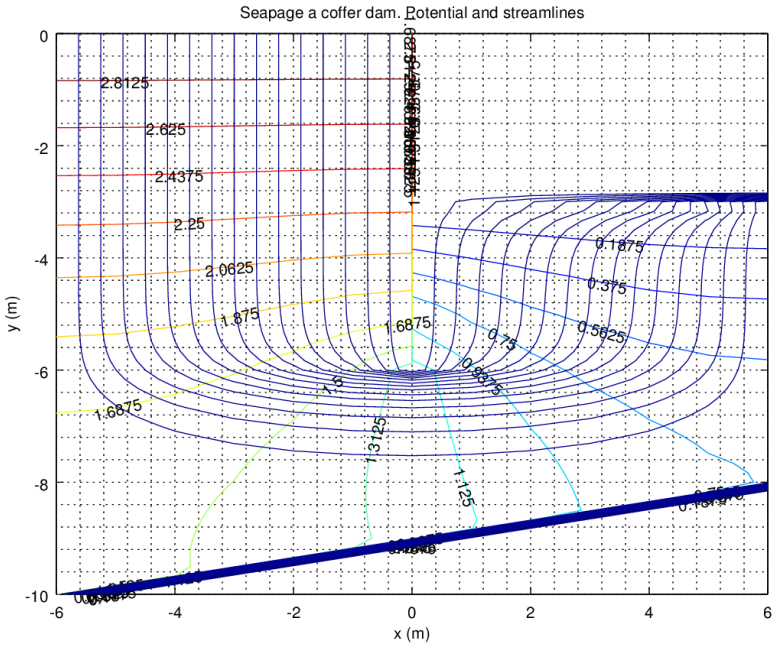
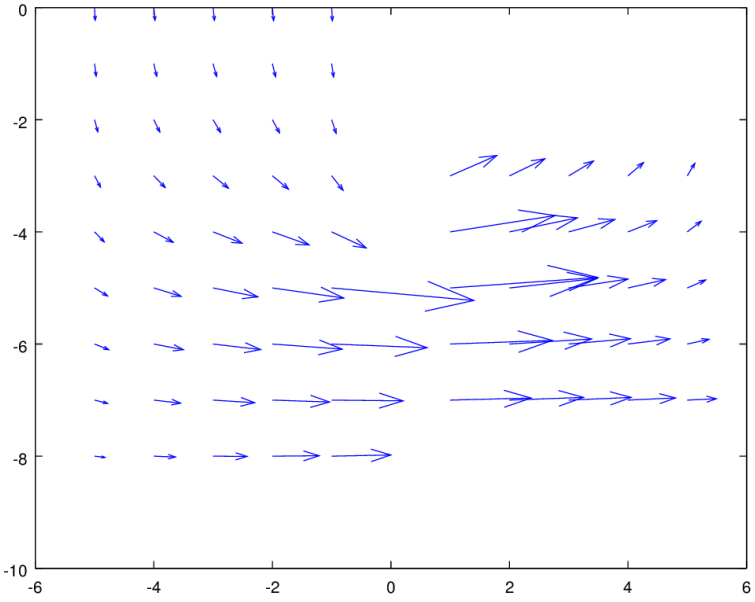


Figure 4.2: Potential lines and stream line

Algorithm 4.1 Velocity vectors



Chapter 5

Calculating Flow Using Finite Element Method

In this chapter we use GeoStudio software to determine the solution of problem in previous chapter using finite element method. We use SEEP/W tool from this set of geotechnical software to solve problem described in previous chapter.

5.1 About SEEP/W

SEEP/W is a finite element CAD software product for analyzing groundwater seepage and excess pore-water pressure dissipation problems within porous materials such as soil and rock. Its comprehensive formulation allows you to consider analyses ranging from simple, saturated steady-state problems to sophisticated, saturated/unsaturated time-dependent problems. SEEP/W can be applied to the analysis and design of geotechnical, civil, hydrogeological, and mining engineering projects.

SEEP/W can model both saturated and unsaturated flow, a feature that greatly broadens the range of problems that can be analyzed. In addition to traditional steady-state saturated flow analysis, the saturated/unsaturated formulation of SEEP/W makes it possible to analyze seepage as a function of time and to consider such processes as the infiltration of precipitation. The transient feature allows you to analyze such problems as the migration of a wetting front and the dissipation of excess pore-water pressure.

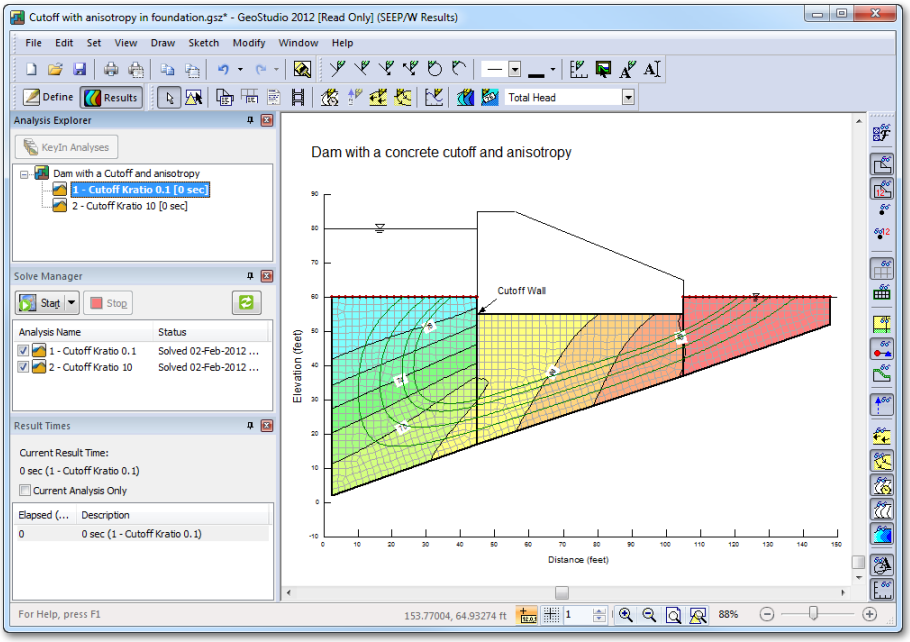


Figure 5.1: GeoStudio SEEP/W software

5.2 Results

Total head and stream lines and velocity vectors are shown in Figure 5.2. Discharge is calculated as $Q = 3.9123e - 6\, m^3/s$.

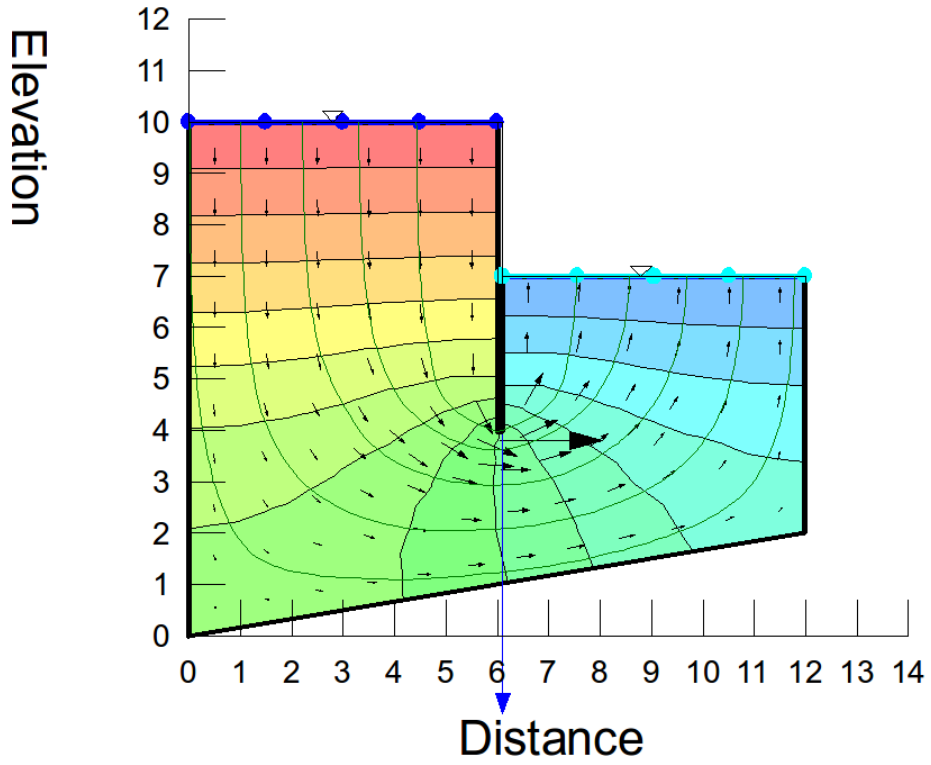


Figure 5.2: Total head, stream line and velocity vectors

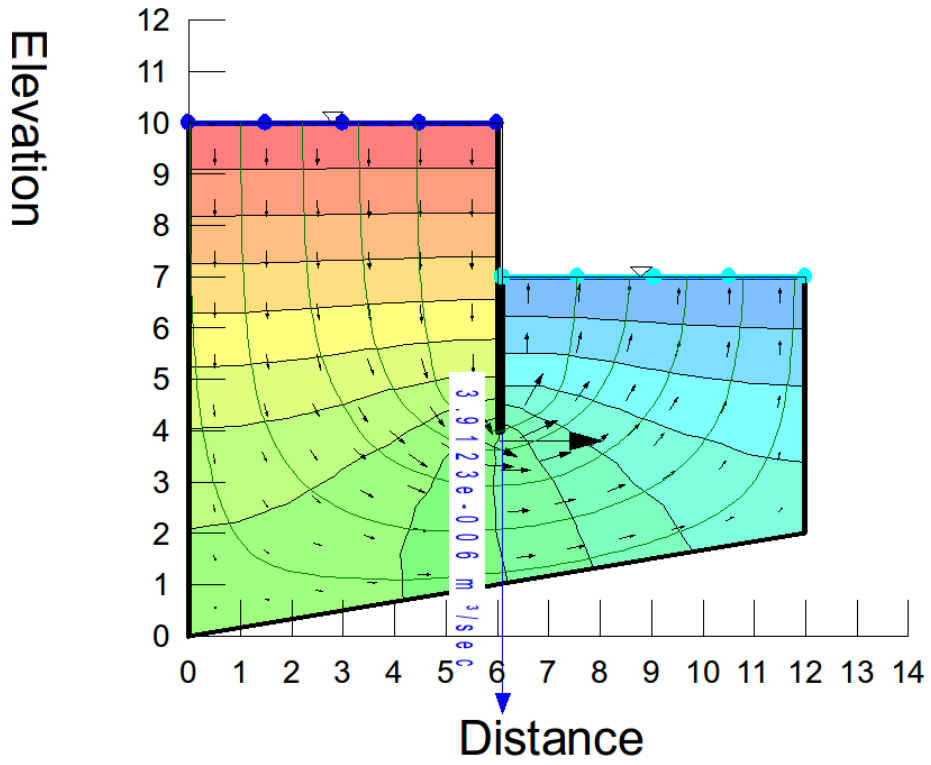


Figure 5.3: Discharge

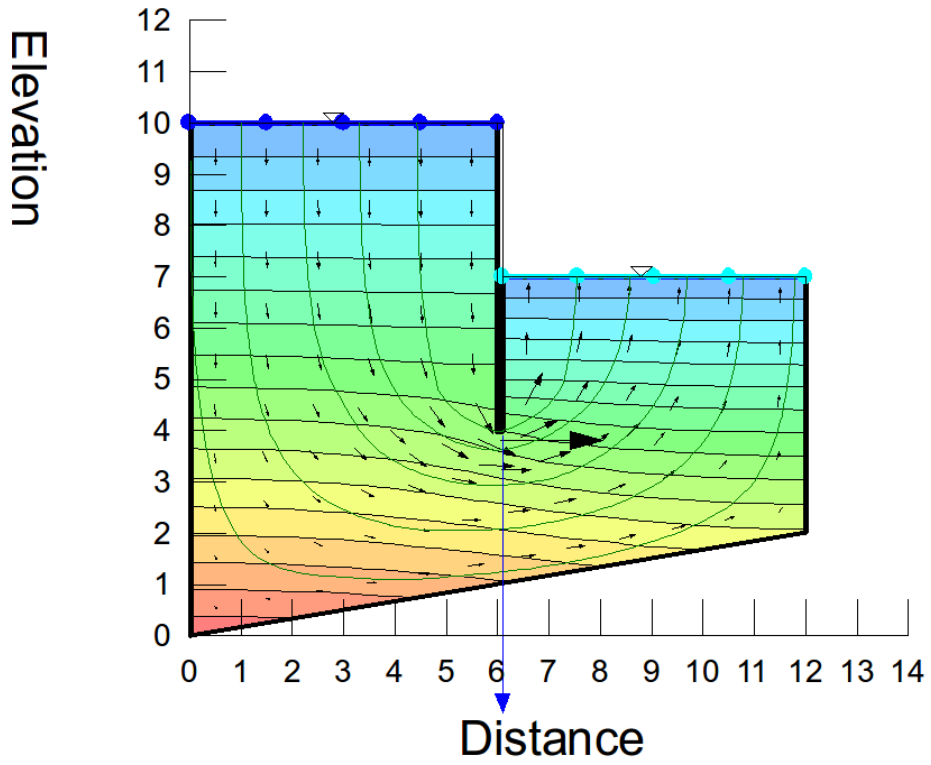


Figure 5.4: Pore pressure