Background material for the Scilab script "CompHydraulicsIII_Script.sce" based on Chapter 5 - from Vreugdenhil, C.B., 1989, "Computational Hydraulics - An Introduction," Springer-Verlag, Berlin.

Material from Chapter 5 - Explicit Finite-Difference Methods
A general explicit method for equation (4.3) is given by equation (5.1). In this equation, α is a parameter that can be adjusted to ensure stability and accuracy. If $\alpha = 1$, the resulting expression is known as the *Lax* method. If $\alpha = 0$, we get the explicit method of Chapter 4. The condition for stability of this method is $\sigma^2 \le \alpha \le 1$, where $\sigma = u \cdot \Delta t / \Delta x$ = Courant number. This is an example of a two-level method.

Function *twoleve.m* was developed to implement the solution using the two-level method scheme presented above. Here is a listing of the function:

```
function [x,t,c,sigma] = twolevel(a,b,Dx,t0,tm,Dt,u,alpha,ci,cb)

% =================================================================
% | This function calculates the concentration of contaminant   |
% | c(j,n) = c(x(j),t(n)), j = 1,2,...,J; n = 1,2,...,N, as     |
% | well as the positions x(j) and time levels t(n) for the     |
% | case of contaminant injection fixo a one-dimensional flow   |
% | domain.  The solution by solving the partial differential   |
% | equation:    dc/dt + u*(dc/dx) + c/T = 0,                    |
% | where u is the flow velocity.  The solution requires         |
% | function ci(x) which provides the initial conditions         |
% | c(x(j),t0) = ci(x(j)), and function cb(t) which provides     |
% | the upstream boundary condition c(a,t(n)) = cb(t(n)).        |
% | The solution is found in the x-domain = [a,b], and time      |
% | domain = [t0,tm].  The x-increment is Dx, and the time       |
% | increment is Dt.  The method used is a general explicit      |
% | method as indicated in equation (5.1), p. 23 from            |
% | Vreugdenhil, C.B., 1989, "Computational                      |
% | Hydraulics," Springer-Verlag, Berlin - Chapter 5.            |
% | alpha parameter for explicit method such that:               |
% | sigma^2 < alpha < 1 for stability                            |
% =================================================================

% Calculate the Courant number, sigma
sigma = u*Dt/Dx;

% Calculate the number of x positions (J) and the number of
% time steps (N), and create vectors x, t, and matrix c:

J = fix((b-a)/Dx) + 1;
N = fix((tm-t0)/Dt) + 1;
x = [a:Dx:b];
t = [t0:Dt:tm];
c = zeros(J,N);

% Load initial conditions

c(:,1) = feval(ci,x); %ci(x)

% Calculation of concentrations

for n = 1:N-1
      % Load boundary conditions
      c(1,n+1) = feval(cb,t(n+1));       %cb(t(n+1)) at u/s
      c(J,n+1) = c(J,n) + u*Dt/Dx*(c(J,n)-c(J-1,n));  % d/s
```

```
        % Calculate concentarations in fixerior pofixs
       for j = 2:J-1
              c(j,n+1) = 0.5*(alpha-sigma)*c(j+1,n)+(1-
alpha)*c(j,n)+0.5*(alpha+sigma)*c(j-1,n);
       end
end

% End function
```

Script *CompHydExIII_Script.m* was put together to perform the solution of the contaminant transport problem with the two-level method described above. The script reproduces Figure 5.1 in page 24 of Vbreugdenhil's book.

```
% Chapter 5 - Explicit Finite-Difference Methods
% 5.1. Two-level methods, eq. (5.1)
%   (c(j,n+1)-0.5*alpha*(c(j+1,n)+c(j-1,n))-(1-alpha)*c(j,n))/Dt +
%   u*(c(j+1,n)-c(j-1,n))/(2*Dx) = 0
% with sigma^2 < alpha < 1 for stability, where sigma = u*Dt/Dx
% sigma is the Courant number.
% An explicit solution for c(j,n+1) is given by:
% c(j,n+1) = 0.5*(alpha-sigma)*c(j+1,n) + (1-alpha)*c(j,n) +
0.5*(alpha+sigma)*c(j-1,n)
% initial conditions function
c0 = 100;
m_file1 = fopen('ci.m','w');
fprintf(m_file1,'function [cci] = ci(x)\r');
fprintf(m_file1,'\nif x==0 \r');
fprintf(m_file1,'\n   cci = %s; \r',num2str(c0));
fprintf(m_file1,'\nelse \r');
fprintf(m_file1,'\n   cci = 0; \r');
fprintf(m_file1,'\nend; \r');
fclose(m_file1);
% boundary conditions function
cb = inline(num2str(c0),'t');
% Solution with alpha = 0.9 and sigma = 0.5
a = 0; b = 20000; t0 = 0; tm = 22000;
Dx = 500; Dt = 500; u = 0.5; alpha=0.9;
% Display input data
disp('Explicit solution to contaminant transport equation');
disp('Solution with alpha = 0.9 and sigma = 0.5');
disp('Input data:')
disp(['a     = ',num2str(a)]);
disp(['b     = ',num2str(b)]);
disp(['t0    = ',num2str(t0)]);
disp(['tm    = ',num2str(tm)]);
disp(['Dx    = ',num2str(Dx)]);
disp(['Dt    = ',num2str(Dt)]);
disp(['u     = ',num2str(u)]);
disp(['alpha = ',num2str(alpha)]);
% Calculate concentration
disp(' ');disp('Calculating concentration');
[x,t,c,sigma]=twolevel(a,b,Dx,t0,tm,Dt,u,alpha,'ci',cb);
disp(['Calculation finished, sigma = ',num2str(sigma)]);
% Plot 3-d solution
figure(1);clf;surf(x,t,c');
xlabel('x');ylabel('t');zlabel('c');
disp('surface plot of concentration shown in Figure 1 - press <return> to
continue');
pause
% Show plots x-vs-t
```

```
disp('concentration vs. position for different times - press <return> to
continue');
figure(2);
for n=4:4:44
   clf;plot(x,c(:,n));
   title(['contaminant transport, n
=',num2str(n)]);xlabel('x(m)');ylabel('c(mg/l)');
   pause
end
pause

% Graph of solution at selected times - figure. 5.1(a), p.24
disp('plot of solution at selected times - press <return> to continue');
figure(3);clf;subplot(1,2,1);
plot(x,c(:,15),'r',x,c(:,30),'m',x,c(:,44),'b');
legend('t=7500 s','t=15000 s','t = 22000 s');
axis([0 20000 0 150]);
title('alpha = 0.9 - sigma = 0.5');xlabel('x(m)');ylabel('c(mg/l)');
pause

% Solution with alpha = 0.9 and sigma = 1.0
c0 = 100;
a = 0; b = 20000; t0 = 0; tm = 22000;
Dx = 500; Dt = 1000; u = 0.5; alpha=0.9;
% Display input data
disp('Explicit solution to contaminant transport equation');
disp('Solution with alpha = 0.9 and sigma = 1.0');
disp('Input data:')
disp(['a     = ',num2str(a)]);
disp(['b     = ',num2str(b)]);
disp(['t0    = ',num2str(t0)]);
disp(['tm    = ',num2str(tm)]);
disp(['Dx    = ',num2str(Dx)]);
disp(['Dt    = ',num2str(Dt)]);
disp(['u     = ',num2str(u)]);
disp(['alpha = ',num2str(alpha)]);
% Calculate concentration
disp(' ');disp('Calculating concentration');
[x,t,c,sigma]=twolevel(a,b,Dx,t0,tm,Dt,u,alpha,'ci',cb);
disp(['Calculation finished, sigma = ',num2str(sigma)]);

% Plot 3-d solution
figure(4);clf;surf(x,t,c');
xlabel('x');ylabel('t');zlabel('c');
disp('surface plot of concentration shown in Figure 4 - press <return> to
continue');
pause

% Show plots x-vs-t
disp('concentration vs. position for different times - press <return> to
continue');
figure(5);
for n=2:2:22
   clf;plot(x,c(:,n));title(['contaminant transport, n =',num2str(n)]);
   xlabel('x(m)');ylabel('c(mg/l)');
   pause
end
pause

% Graph of solution at selected times - figure. 5.1(a), p.24
disp('plot of solution at selected times - press <return> to continue');
figure(3);;subplot(1,2,2);
plot(x,c(:,8),'r',x,c(:,15),'m',x,c(:,22),'b');
```

```
legend('t=8000 s','t=15000 s','t = 22000 s');
axis([0 20000 0 150]);
title('alpha = 0.9 - sigma = 1.0');xlabel('x(m)');ylabel('c(mg/l)');

% End script
```

Function *ci.m* gets created by the script:

```
function [cci] = ci(x)
if x==0
   cci = 100;
else
   cci = 0;
end;
```