

Background material for the Matlab script "CompHydraulicsI_Script.sce" based on Chapters 2 and 3 - from Vreugdenhil, C.B., 1989, "Computational Hydraulics - An Introduction," Springer-Verlag, Berlin.

Material from Chapter 2 - Water Quality in a Lake

Consider a lake with a single input discharge, Q_i , carrying a contaminant (say, BOD) at a concentration c_i . The average concentration of contaminant in the lake is c . The lake has an output discharge of Q_o at a concentration c , the same as the lake concentration. We can also include in the balance an amount L representing the amount of contaminant discharged directly in the lake (say, by dumping contaminant). The volume of the lake is V , and the rate of degradation is given by Vc/T_r , where T_r is a degradation time scale. The mass of contaminant in the lake at any given time is given by Vc . Thus, the rate of change of that mass is given by the equation

$$\frac{d}{dt}(Vc) = Q_i c_i + L - Q_o c - \frac{Vc}{T_r}.$$

For equal inflow and outflow volumes $Q_i=Q_o$, the equation simplifies to

$$\frac{d}{dt}(Vc) = Q(c_i - c) + L - \frac{Vc}{T_r}.$$

What we need to know to solve this equation:

- The initial concentration, c_o
- The river discharge, Q
- The inflow concentration, c_i
- The magnitude of the of the direct discharge, L

If the system reaches an equilibrium condition, then $dc/dt = 0$, and the equation produces the equilibrium concentration

$$c_e = \frac{L + Q_i c_i}{Q_o + \frac{V}{T_r}}.$$

With this value, we can write the governing equation as

$$\frac{dc}{dt} + \frac{c}{T} = \frac{c_e}{T},$$

where

$$T = \frac{V}{Q_i + \frac{V}{T_r}}.$$

Notice that if $Q_j = 0$, $T = T_r$, i.e., the degradation time scale. On the other hand, if $T_r \rightarrow \infty$, $T = V/Q_j$, the flushing time.

With $c(0) = c_o$, an analytical solution to the governing equation is

$$c(t) = c_e + (c_o - c_e)e^{-t/T}.$$

Material from Chapter 3 - Numerical Solution for Box Model

Numerical solutions to the governing ODE can be obtained by using Matlab's pre-defined function `ode45` (use: `>> help ode45` for details), or by using the user-defined function `Euler1`, which implements Euler's first-order explicit numerical solution. NOTE: function `ode45` implements a fourth-fifth-order Runge-Kutta numerical solution for the ODE.s

Euler's method is the basis for equation (3.1) in Vreugdenhill's book:

$$c_{j+1} = c_j + \frac{\Delta t}{T}(c_e - c_j).$$

Equation (3.2) is used to analyze the *stability* of Euler's method for this equation. The analysis indicate that the condition for stability is

$$\Delta t/T < 2.$$

Page 14 in Vreugdenhill's book shows the following finite-difference scheme for the numerical solution of the equation:

$$\frac{c_{j+1} - c_j}{\Delta t} = \theta \frac{(c_e)_{j+1} - c_{j+1}}{T_{j+1}} + (1 - \theta) \frac{(c_e)_j - c_j}{T_j}$$

from which an explicit solution can be found

$$c_{j+1} = \frac{c_e \Delta t + c_j (T + (1 - \theta) \Delta t)}{T + \theta \Delta t}.$$

An implicit method for the differential equation results in writing the following equations:

$$a_{j,j-1}c_{j-1} + a_{jj}c_j + a_{j,j+1}c_{j+1} = b_j, \quad j = 1, 2, \dots, n-1$$

$$a_{j,j-1} = 0, \quad a_{jj} = \frac{(1 - \theta)\Delta t}{T - 1}, \quad a_{j,j+1} = \frac{\theta \cdot \Delta t}{T + 1}, \quad b_j = \frac{\Delta t \cdot c_e}{T}, \quad a_{11} = 1$$

These equations can be re-written in terms of a matrix equation of the form $A \cdot c = b$, and solved using Matlab's left-division, i.e., $c = A \setminus b$. The following Matlab commands solve the implicit

equations for values of $\theta = 0.6, 0.7, 0.8, 0.9$, and 1.0 , and plot results for this explicit solution:

The following Matlab commands produce graphs of c vs t for different values of $\theta = 0.5, 0.6, 0.7, 0.8, 0.9$, and 1.0 , for this explicit solution:

Matlab script *CompHydExl_Script.m* shows the exact and numerical solutions with different methods for the problem of water quality in a lake. The data used in the solution is the following:

```
Q = 86400 m^3/day
V = 600000 m^3
L = 10307520 kg/day
ci = 5 kg/m^3
ce = 37.4983 kg/m^3
T = 2.095 days
```

The specific calculation or graph produced by the script is properly commented in the script file:

```
% Based on Chapter 2 - Water Quality in a Lake from Vreugdenhil, C.B.,
% 1989, "Computational Hydraulics - An Introduction," Springer-Verlag,
% Berlin
% Concentration of contaminant in a lake
Q = 86400; % m^3/day, input and output discharge
V = 6e5; % m^3, volume
Tr=3; % days, degradation time scale
L= 119.3*86400; % kg/day, contaminant load
ci = 5; % mg/l = kg/m^3, input contaminant
concentration
ce = (L+Q*ci)/(Q+V/Tr); % mg/l = kg/m^3, equilibrium concentration
T = V/(Q+V/Tr); % day, relaxation time
disp('Water quality in a lake');
disp('=====');
disp(['Q = ',num2str(Q), ' m^3/day']);
disp(['V = ',num2str(V), ' m^3']);
disp(['L = ',num2str(L), ' kg/day']);
disp(['ci = ',num2str(ci), ' kg/m^3']);
disp(['ce = ',num2str(ce), ' kg/m^3']);
disp(['T = ',num2str(T), ' days']);
pause

% Exact solution for different initial concentration c0
% See eq. (2.8), Fig. 2.2, p. 7
disp(' ');disp('Exact solution -- see figure 1');
t = [0:0.5:30];
s1 = 'ce+(c0-ce)*exp(-t/T) '
s1 = strrep(s1,'ce',num2str(ce));
s1 = strrep(s1,'T',num2str(T));
c0 = 0.0*ce; s2 = strrep(s1,'c0',num2str(c0));c = inline(s2,'t');cc1 =
c(t);
c0 = 0.2*ce; s2 = strrep(s1,'c0',num2str(c0));c = inline(s2,'t');cc2 =
c(t);
c0 = 0.5*ce; s2 = strrep(s1,'c0',num2str(c0));c = inline(s2,'t');cc3 =
c(t);
```

```

figure(1);plot(t,cc1,'-',t,cc2,'--',t,cc3,'-.');legend('c0/ce =
0.0','c0/ce = 0.2','c0/ce = 0.5',4);
title('Exact solution');xlabel('t(day)');ylabel('c(mg/l)')
pause

% Numerical solution using MATLAB function 'ode45'
% f001(t,c)=(ce-c)/T -- inline function - solution for different
% values of c0:
disp(' ');disp('Numerical solution with ode45 -- see figure 2');
t0 = 0; tf = 30; Dt = 3; tRange = [t0 tf];
s1 = '(ce-c)/T'; s1 = strrep(s1,'T',num2str(T));s1 =
strrep(s1,'ce',num2str(ce));
clear c; f001 = inline(s1,'t','c');
c0 = 0.0*ce; [t1,c1] = ode45(f001,tRange,c0);
c0 = 0.2*ce; [t2,c2] = ode45(f001,tRange,c0);
c0 = 0.5*ce; [t3,c3] = ode45(f001,tRange,c0);
figure(2);plot(t1,c1,'-',t2,c2,'--',t3,c3,'-.');legend('c0/ce =
0.0','c0/ce = 0.2','c0/ce = 0.5',4);
title('Numerical solution - function ode45 - Dt = 3
days');xlabel('t(day)');ylabel('c(mg/l)')
pause

% Numerical solution using different initial conditions with user-
defined function "Euler1" - solution for different values of c0:
disp(' ');disp('Numerical solution with Euler1 -- see figure 3');
[ttt,ccc1]=Euler1(0,0*ce,tf,Dt,f001);
[ttt,ccc2]=Euler1(0,0.2*ce,tf,Dt,f001);
[ttt,ccc3]=Euler1(0,0.5*ce,tf,Dt,f001);
figure(3);plot(ttt,ccc1,'-',ttt,ccc2,'--',ttt,ccc3,'-.');legend('c0/ce
= 0.0','c0/ce = 0.2','c0/ce = 0.5',4);
title('Numerical solution (Euler1) - different c0 - Dt = 3
days');xlabel('t(day)');ylabel('c(mg/l)')
pause

% Numerical solution for different time steps with MATLAB function
'ode45' - illustrates stability of Runge-Kutta 4/5 method
disp(' ');disp('Numerical solution with ode45, different time steps --
see figure 4');
Dt = 0.50*T; t1 = [0:Dt:10*T]; [t1,c1] = ode45(f001,t1,0.2*ce);
Dt = 0.30*T; t2 = [0:Dt:10*T]; [t2,c2] = ode45(f001,t2,0.2*ce);
Dt = 0.03*T; t3 = [0:Dt:10*T]; [t3,c3] = ode45(f001,t3,0.2*ce);
figure(4); plot(t1,c1,'-',t2,c2,'--',t3,c3,'-.');
legend('Dt/T=0.50','Dt/T=0.30','Dt/T=0.03',4);
title('Numerical solution (ode45) - different Dt -- c0/ce =
0.2');xlabel('t(day)');ylabel('c(mg/l)')
pause

% Numerical solution for different time steps with user-defined
function 'Euler1' - illustrates stability of Euler's 1-st order method
disp(' ');disp('Numerical solution with Euler1, different time steps --
see figure 5');
[t1,c1] = Euler1(0.0,0.2*ce,10*T,0.5*T,f001);
[t2,c2] = Euler1(0.0,0.2*ce,10*T,0.3*T,f001);
[t3,c3] = Euler1(0.0,0.2*ce,10*T,0.03*T,f001);
figure(5); plot(t1,c1,'-',t2,c2,'--',t3,c3,'-.');
legend('Dt/T=0.50','Dt/T=0.30','Dt/T=0.03',4);

```

```

title('Numerical solution (Euler1) - different Dt -- c0/ce =
0.2');xlabel('t(day)');ylabel('c(mg/l)')
pause

% Numerical solution for different time steps with user-defined
function 'Euler1' - Plotting dimensionless concentration
% cc = c/ce - vs - dimensionless time tt = t/T
disp(' ');disp('Numerical solution with Euler1, different time steps --
see figure 6');
[t1,c1] = Euler1(0.0,0.2*ce,6*T,3*T,f001);
tt1=t1/T;cc1=c1/ce;
[t2,c2] = Euler1(0.0,0.2*ce,9*T,1.5*T,f001);
tt2=t2/T;cc2=c2/ce;
[t3,c3] = Euler1(0.0,0.2*ce,9*T,0.03*T,f001);
tt3=t3/T;cc3=c3/ce;
figure(6);plot(tt1,cc1,'-',tt2,cc2,'--',tt3,cc3,'-.');
legend('Dt/T=3','Dt/T=1.5','Dt/T=0.03',4)
title('Numerical solution (Euler1)- dimensionless - different Dt -
c0/ce = 0.2');xlabel('t/T');ylabel('c/ce')
pause

% Modify equation to show a sudden release as in Figure 3.3, p. 13
% First, create m-file function 'f002.m' used for the numerical
% solution with 'ode45':
disp(' ');disp('Results from figure 3.3 -- see figure 7');
m_file1 = fopen('f002.m','w');
fprintf(m_file1,'function [Df] = f002(t,c)\r');
s1 = '(ce-c)/T'; s1 = strrep(s1,'T',num2str(T));
sA = strrep(s1,'ce',num2str(ce));sB = strrep(s1,'ce','100');
fprintf(m_file1,'\nif t<1 | t>3 \r');
fprintf(m_file1,'\n    Df = %s; \r',sA);
fprintf(m_file1,'\nelse \r');
fprintf(m_file1,'\n    Df = %s; \r',sB);
fprintf(m_file1,'\nend; \r');
fclose(m_file1);
% Calculate solution of 0 < t < 5
t = [0:0.1:5]; [t,c] = ode45('f002',t,0);
% Next, create m-file function 'cee.m' to show the concentration
% variation
m_file1 = fopen('cee.m','w');
fprintf(m_file1,'function [cc] = cee(t)\r');
fprintf(m_file1,'\nif t<1 | t>3 \r');
fprintf(m_file1,'\n    cc = %s; \r',num2str(ce));
fprintf(m_file1,'\nelse \r');
fprintf(m_file1,'\n    cc = %s; \r',num2str(100));
fprintf(m_file1,'\nend; \r');
fclose(m_file1);
% Evaluate function 'cee.m':
cc = []; for k=1:length(t), cc=[cc cee(t(k))]; end;
figure(7);plot(t,c,'-',t,cc,'--');legend('ce','c(t)'); axis([0 5 0
110]);
title('Numerical solution - function ode45 - modified
equation');xlabel('t/T');ylabel('c/ce')
pause

% Page 14 - Explicit version of the implicit method for the equation:
%
dc/dt = (ce-c)/T,

```

```

% keeping both T and ce constant in the expression of page 14.
clear;
disp(' ');disp('Explicit method of page 14 -- see figures 8 and 9');
% Use same data as above with Dt = 0.1:
Q=86400;V=6e5;Tr=3;L=10307520;ci=5;ce=37.49;T=2.094;Dt = 0.1;
% Calculation for theta = 0.5
theta=0.5;t=[0:Dt:5];n=length(t);c1=zeros(1,n);c1(1)=ci;
for j = 1:n-1
    c1(j+1)=(ce*Dt+c1(j)*(T+(1-theta)*Dt))/(T+theta*Dt);
end;
figure(8);clf;plot(t,c1); legend('theta=0.5',4);
title('Explicit method from page
14');xlabel('t(days)');ylabel('c(mg/l)');
pause
% Calculation for theta = 0.6
theta=0.6;t=[0:Dt:5];n=length(t);c2=zeros(1,n);c2(1)=ci;
for j = 1:n-1
    c2(j+1)=(ce*Dt+c2(j)*(T+(1-theta)*Dt))/(T+theta*Dt);
end;
figure(8);clf;plot(t,c2); legend('theta=0.6',4);
title('Explicit method from page
14');xlabel('t(days)');ylabel('c(mg/l)');
pause
% Calculation for theta = 0.7
theta=0.7;t=[0:Dt:5];n=length(t);c3=zeros(1,n);c3(1)=ci;
for j = 1:n-1
    c3(j+1)=(ce*Dt+c3(j)*(T+(1-theta)*Dt))/(T+theta*Dt);
end;
figure(8);clf;plot(t,c3); legend('theta=0.7',4);
title('Explicit method from page
14');xlabel('t(days)');ylabel('c(mg/l)');
pause
% Calculation for theta = 0.8
theta=0.8;t=[0:Dt:5];n=length(t);c4=zeros(1,n);c4(1)=ci;
for j = 1:n-1
    c4(j+1)=(ce*Dt+c4(j)*(T+(1-theta)*Dt))/(T+theta*Dt);
end;
figure(8);clf;plot(t,c4); legend('theta=0.8',4);
title('Explicit method from page
14');xlabel('t(days)');ylabel('c(mg/l)');
pause
% Calculation for theta = 0.9
theta=0.9;t=[0:Dt:5];n=length(t);c5=zeros(1,n);c5(1)=ci;
for j = 1:n-1
    c5(j+1)=(ce*Dt+c5(j)*(T+(1-theta)*Dt))/(T+theta*Dt);
end;
figure(8);clf;plot(t,c5); legend('theta=0.9',4);
title('Explicit method from page
14');xlabel('t(days)');ylabel('c(mg/l)');
pause
% Calculation for theta = 1.0
theta=1.0;t=[0:Dt:5];n=length(t);c6=zeros(1,n);c6(1)=ci;
for j = 1:n-1
    c6(j+1)=(ce*Dt+c6(j)*(T+(1-theta)*Dt))/(T+theta*Dt);
end;
figure(8);clf;plot(t,c6); legend('theta=1.0',4);

```

```

title('Explicit method from page
14');xlabel('t(days)');ylabel('c(mg/l)');
pause

% Plot of the solutions for different values of theta
figure(9);clf;
plot(t,c1,'-',t,c2,'--',t,c3,'-.',t,c4,'+',t,c5,'o',t,c6,'v');
legend('theta=0.5','theta=0.6','theta=0.7','theta=0.8','theta=0.9','the
ta=1.0',2);
title('Explicit method from page
14');xlabel('t(days)');ylabel('c(mg/l)');
pause

% Implicit method from page 14:
disp(' ');disp('Implicit method of page 14 -- see figures 10 and 11');
Q=86400;V=6e5;Tr=3;L=10307520;ci=5;ce=37.49;T=2.094;Dt = 0.1;
% Calculation for theta = 0.5
theta=0.5;t=[0:Dt:10];n=length(t);A=zeros(n-1,n-1);b=zeros(n-1,1);
for j = 1:n-1
    A(j,j) = (1-theta)*Dt/T-1;A(j,j+1) = theta*Dt/T+1;b(j) = Dt*ce/T;
end;
A = [zeros(1,n);A];A(1,1) = 1.0;b = [ci;b];c1 = A\b;
figure(10);clf;plot(t,c1); legend('theta=0.5',4);
title('Implicit method from page
14');xlabel('t(day)');ylabel('c(mg/l)');
pause
% Calculation for theta = 0.6
theta=0.6;t=[0:Dt:10];n=length(t);A=zeros(n-1,n-1);b=zeros(n-1,1);
for j = 1:n-1
    A(j,j) = (1-theta)*Dt/T-1;A(j,j+1) = theta*Dt/T+1;b(j) = Dt*ce/T;
end;
A = [zeros(1,n);A];A(1,1) = 1.0;b = [ci;b];c2 = A\b;
figure(10);clf;plot(t,c2); legend('theta=0.6',4);
title('Implicit method from page
14');xlabel('t(day)');ylabel('c(mg/l)');
pause
% Calculation for theta = 0.7
theta=0.7;t=[0:Dt:10];n=length(t);A=zeros(n-1,n-1);b=zeros(n-1,1);
for j = 1:n-1
    A(j,j) = (1-theta)*Dt/T-1;A(j,j+1) = theta*Dt/T+1;b(j) = Dt*ce/T;
end;
A = [zeros(1,n);A];A(1,1) = 1.0;b = [ci;b];c3 = A\b;
figure(10);clf;plot(t,c3); legend('theta=0.7',4);
title('Implicit method from page
14');xlabel('t(day)');ylabel('c(mg/l)');
pause
% Calculation for theta = 0.8
theta=0.8;t=[0:Dt:10];n=length(t);A=zeros(n-1,n-1);b=zeros(n-1,1);
for j = 1:n-1
    A(j,j) = (1-theta)*Dt/T-1;A(j,j+1) = theta*Dt/T+1;b(j) = Dt*ce/T;
end;
A = [zeros(1,n);A];A(1,1) = 1.0;b = [ci;b];c4 = A\b;
figure(10);clf;plot(t,c4); legend('theta=0.8',4);
title('Implicit method from page
14');xlabel('t(day)');ylabel('c(mg/l)');
pause

```

```

% Calculation for theta = 0.9
theta=0.9;t=[0:Dt:10];n=length(t);A=zeros(n-1,n-1);b=zeros(n-1,1);
for j = 1:n-1
    A(j,j) = (1-theta)*Dt/T-1;A(j,j+1) = theta*Dt/T+1;b(j) = Dt*ce/T;
end;
A = [zeros(1,n);A];A(1,1) = 1.0;b = [ci;b];c5 = A\b;
figure(10);clf;plot(t,c5); legend('theta=0.9',4);
title('Implicit method from page
14');xlabel('t(day)');ylabel('c(mg/l)');
pause
% Calculation for theta = 1.0
theta=1.0;t=[0:Dt:10];n=length(t);A=zeros(n-1,n-1);b=zeros(n-1,1);
for j = 1:n-1
    A(j,j) = (1-theta)*Dt/T-1;A(j,j+1) = theta*Dt/T+1;b(j) = Dt*ce/T;
end;
A = [zeros(1,n);A];A(1,1) = 1.0;b = [ci;b];c6 = A\b;
figure(10);clf;plot(t,c6); legend('theta=1.0',4);
title('Implicit method from page
14');xlabel('t(day)');ylabel('c(mg/l)');
pause

% Plot of the solutions for different values of theta
figure(11);clf;
plot(t,c1,'-',t,c2,'--',t,c3,'-.',t,c4,'+',t,c5,'o',t,c6,'v');
legend('theta=0.5','theta=0.6','theta=0.7','theta=0.8','theta=0.9','the
ta=1.0',4);
title('Implicit method from page
14');xlabel('t(days)');ylabel('c(mg/l)');
% End of script

```

Function *Euler1.m* is shown next:

```

function [x,y] = Euler1(x0,y0,xn,Dx,g)
% Euler 1st order method solving ODE
% dy/dx = g(x,y), with initial
% conditions y=y0 at x = x0. The
% solution is obtained for x = [x0:Dx:xn]
% and returned in y
ymaxAllowed = 1e+100;
x = [x0:Dx:xn]; y = zeros(1,length(x)); n = length(y); y(1) = y0;
for j = 1:n-1
    y(j+1) = y(j) + Dx*feval(g,x(j),y(j));
    if y(j+1) > ymaxAllowed
        disp('Euler 1 - WARNING: underflow or overflow');
        disp('Solution sought in the following range:');
        disp([x0 Dx xn]);
        disp('Solution evaluated in the following range:');
        disp([x0 Dx x(j)]);
        n = j; x = x(1,1:n); y = y(1,1:n);
        break;
    end;
end;
% End function Euler1

```

The script generates two functions needed to reproduce Figure 3.3., namely:

```
function [Df] = f002(t,c)
if t<1 | t>3
    Df = (37.4983-c)/2.095;
else
    Df = (100-c)/2.095;
end;
```

and,

```
function [cc] = cee(t)
if t<1 | t>3
    cc = 37.4983;
else
    cc = 100;
end;
```

Use

```
» CompHydExI_Script
```

to run the script.

Results are shown in the main Matlab interface as well as in Figures 1 through 11.

This document was prepared by Gilberto E. Urroz, Associate Professor, Department of Civil and Environmental Engineering, Utah State University, on September 15, 2002. The solutions are based on the book Vreugdenhil, C.B., 1989, "Computational Hydraulics - An Introduction," Springer-Verlag, Berlin.