SIK2063 Transportprosesser, høst 2000

Solution: Unsteady diffusion into slab

Problem: Solve eq. 2.3-5 ($\frac{\partial C}{\partial t} = D\frac{\partial^2 C}{\partial z^2}$) in Cussler numerically using the discretization routines *dss020* and *dss042* and the differential equation solver *ode15s*. Solve for both the boundary condition given in eqs. 2.3-6–8 and for the "infinite" BC given as $\frac{\partial C}{\partial z} = 0$.

Programs with boundary condition $C_{end} = 0$

main.m

```
% Main program to solve unsteady diffusion into slab
% --------------------------------------------------

% Initializing
% ------------
clear all
global C10 Cend D z0 zend m

m     = 20;   % Number of discretization points
D     = 1e-9; % Diffusion coefficient
C10   = 10;   % Concentration at interface
C0    = 0*ones(1,m); % Initial values for conc. in slab for all disc. points
Cend  = 0;    % The conc. at the inner position remains constant
z0    = 0;    % Position at interface
zend  = 1e-3; % Position at end [m]
z     = [0:(zend-z0)/(m+1):zend]; % Vector containing all dicr. positions
                                  % and the end points. i.e. m+2 points
t0    = 0;
tend  = 10;   % Final time [s]

% Solving the differential equation
% ---------------------------------
[t,y] = ode15s('tder',[t0 tend],C0);

% Postprosessing of data
% ----------------------
% y contains only the internal discr. points, so we must now add the
% interface and the bulk conc. to y
s     = size(t);
```

```
nt    = s(1);    % Number of time steps
y     = [C10*ones(nt,1),y,Cend*ones(nt,1)];

yend  = y(nt,:); % The final conc. profile
Ca    = C10 + ([C10 C0 Cend] - C10).*erf(z./sqrt(4*D*tend)); % Analytic sol.

% Plotting of results
% -------------------
plot(z,y(nt,:),'.',z,Ca,'-')
xlabel('position [m]')
ylabel('Conc. [mol/m^3]')
legend('Numerical','Analytical')
```

tder.m

```
function dCdt = tder(t,C)

global C10 Cend D z0 zend m

% Since both boundary conditions are given as a function value which is fixed,
% this value must be held outside the intefration and serves as a BC on which
% the derivatives can be based. So we expand the variable vector to also
% contain C10 and Cend
Cmod  = [C10; C; Cend]';
dCdz  = dss020(z0,zend,m+2,Cmod,1);        % 1. order derivative
dCdz2 = dss042(z0,zend,m+2,Cmod,dCdz,1,1); % 2. order derivative

dCdt  = D*dCdz2;     % Calculating the time derivative of C-vector
dCdt  = dCdt(2:m+1)';% Reducing the vector to the m points in the "center" of
                     % the vector. We also convert it to a column vector
                     % to meet ode15s specifications
```

Programs with boundary condition $\frac{\partial C}{\partial z} = 0$ at $z = z_{end}$

main.m

```
% Main program to solve unsteady diffusion into slab
% -------------------------------------------------

% Initializing
% ------------
clear all
```

```
global C10 Cend D z0 zend m

m     = 20;   % Number of discretization points
D     = 1e-9; % Diffusion coefficient
C10   = 10;   % Concentration at interface
C0    = 0*ones(1,m); % Initial values for conc. in slab for all disc. points
Cend  = 0;    % The conc. at the inner position remains constant
z0    = 0;    % Position at interface
zend  = 1e-3; % Position at end [m]
z     = [0:(zend-z0)/m:zend]; % Vector containing all discr. positions,
                              % i.e. m+1 points
t0    = 0;
tend  = 10;   % Final time [s]

% Solving the differential equation
% ---------------------------------
[t,y] = ode15s('tder',[t0 tend],C0);

% Postprosessing of data
% ----------------------
% y contains only the internal discr. points, so we must now add the
% interface conc. to y
s     = size(t);
nt    = s(1);    % Number of timesteps
y     = [C10*ones(nt,1),y];

yend  = y(nt,:); % The final conc. profile
Ca    = C10 + ([C10 C0] - C10).*erf(z./sqrt(4*D*tend)); % Analytic sol.

% Plotting of results
% -------------------
plot(z,y(nt,:),'.',z,Ca,'-')
xlabel('position [m]')
ylabel('Conc. [mol/m^3]')
legend('Numerical','Analytical')
```

tder.m

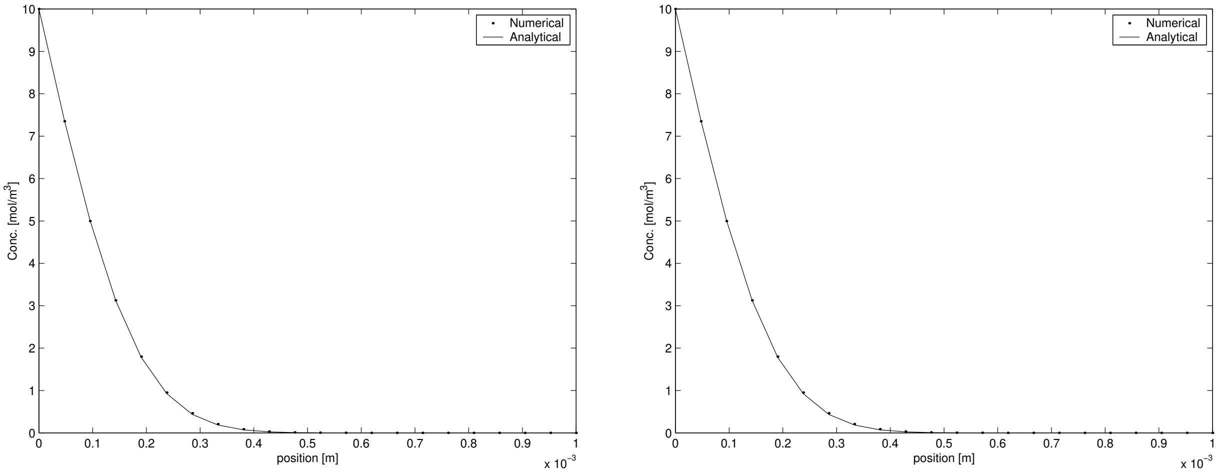
```
function dCdt = tder(t,C)

global C10 Cend D z0 zend m

% Since one of the  boundary conditions is given as a function value which is
```

```
% fixed, this value must be held outside the intefration and serves as a BC on
% which the derivatives can be based. So we expand the variable vector to also
% contain C10
Cmod    = [C10; C]';
dCdz    = dss020(z0,zend,m+1,Cmod,1);      % 1. order derivative
dCdz(m) = 0;           % Setting the 1. derivative at zend to 0
dCdz2   = dss042(z0,zend,m+1,Cmod,dCdz,1,2); % 2. order derivative

dCdt    = D*dCdz2;     % Calculating the time derivative of C-vector
dCdt    = dCdt(2:m+1)';% Reducing the vector to m points by removing the
                       % first element. We also convert it to a column vector
                       % to meet ode15s specifications
```



A B

Figure 1: Resulting plot from the programs. A) with boundary condition $C_{end} = 0.$ and B) with boundary condition $\frac{\partial C}{\partial z} = 0$ at $z = z_{end}$.