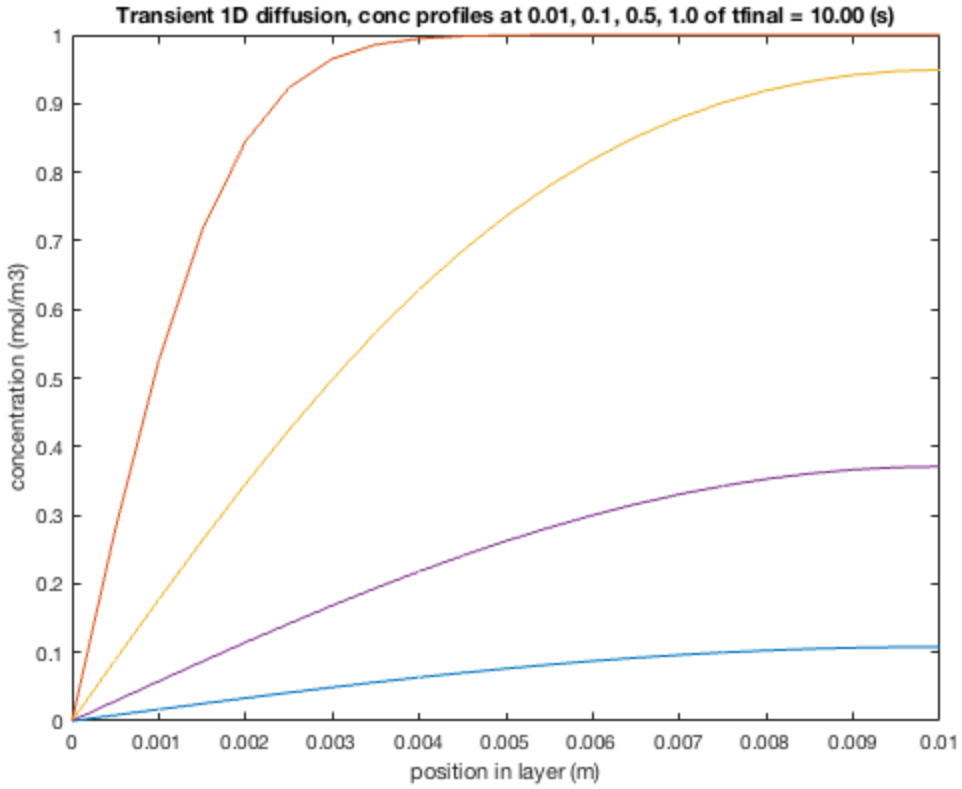
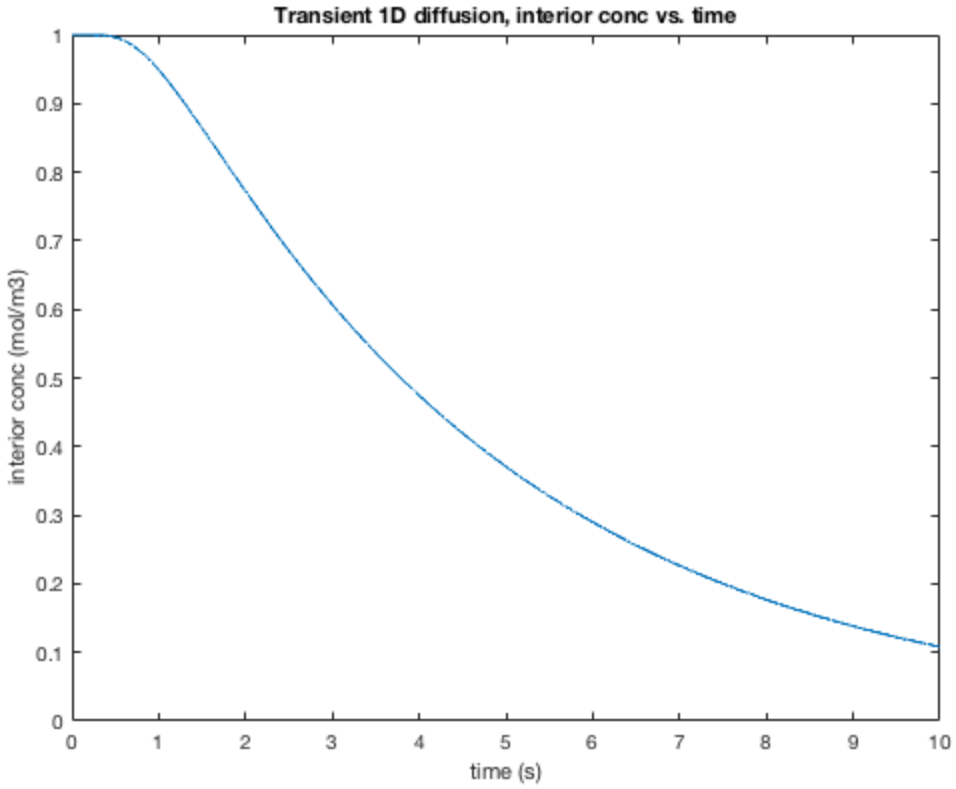
```
% Transient 1D diffusion, uniform layer, one side impermeable,
% as in a medical drug patch, for example.
% Demonstrate a finite-difference method of numerical solution

% ALSO SEE notes attached starting p. 5

% ReactorLab.net

% SUGGSTIONS: Vary n and lambda values to get good approximation to
% analytical solution for this case, which has an analytical solution.

% Numerical methods are most useful for problems not having available
% analytical solution, e.g., conc at layer face varies unpredictably
% or complex variation of D or initial conc within layer.

% OTHER THINGS TO DO: compute rate of diffusion out of layer vs. time;
% compute fraction diffusing component left in layer vs. time.

clear all
fprintf('------------------- \n') % separates runs in command window

% BEGIN USER INPUTS

D = 1.0e-5; % (m2/s), diffusion coefficient
cI = 1; % (mol/m3), initial concentration
c0 = 0; % new conc at layer "face" x = 0 for t>0
L = 0.01; % (m), length of layer, impermeable on one side, "interior"
n = 20; % integer number divisions of L
tfinal = 10; % (s), final time

% lambda = D*dt/dx^2 is a dimensionless diffusion coefficient
% require lambda <= 0.25 for numerical stability

lambda = 0.05;

% END USER INPUTS

% compute time step size dt (s) to get desired lambda
dt = lambda*(L/n)^2/D % where dx = (L/n), length step

% Matlab array indexes are integers starting at 1
% c is 2D concentration array that saves all position and time data
% c(i,j), where i is position index, and j is time index
% c(1,1) is at x = 0, t = 0
% c(n+1,1) is at x = L, t = 0

% specify initial conditions

c(1:n+1,1) = cI;
t(1) = 0;
j = 1; % time index value at t = 0
```

```
jmax = 1+round(tfinal/dt); % max value of time index j
tfinal = (jmax-1) * dt; % recompute tfinal using integer jmax
m = n+1; % index for center node to match notes

% step in time

while t(j) <= tfinal

    % exterior node
    c(1,j+1) = c0;

    % interior nodes
    for i = 2:n
        c(i,j+1) = c(i,j) + lambda*(c(i+1,j)-2*c(i,j)+c(i-1,j));
    end

    % center node at m = n+1
    c(m,j+1) = c(m,j) + lambda*(2*c(m-1,j)-2*c(m,j));

    t(j+1) = t(j) + dt; % update time array
    j = j+1; % increment time index

end

fprintf('number of time steps = %i \n',jmax-1)

fprintf('interior conc at final time = %6.4f \n',c(m,jmax))

x = linspace(0,L,n+1); % position in layer for plot

plot(x,c(1:n+1,jmax));
hold on

plot(x,c(1:n+1,round( 0.01 * jmax)));
plot(x,c(1:n+1,round( 0.1 * jmax)));
plot(x,c(1:n+1,round( 0.5 * jmax)));

tt = sprintf('Transient 1D diffusion, conc profiles at 0.01, 0.1, 0.5,
 1.0 of tfinal = %4.2f (s)',tfinal);

title(tt);
ylabel('concentration (mol/m3)')
xlabel('position in layer (m)')
hold off

figure(2)
plot(t,c(m,1:jmax))
axis([0 10 0 1])
title('Transient 1D diffusion, interior conc vs. time')
ylabel('interior conc (mol/m3)')
xlabel('time (s)')

-------------------
```

```
dt =

    0.0012

number of time steps = 8000
interior conc at final time = 0.1081
```





Published with MATLAB® R2016a

Numerical Integration of Differential Equations, PART B
by Richard K. Herz <herz@ucsd.edu>

Finite Difference Method for Integration of PDE's

These notes will consider integration of the following parabolic PDE using the finite difference method.

$$\frac{\partial c}{\partial t} = D \frac{\partial^2 c}{\partial x^2}$$

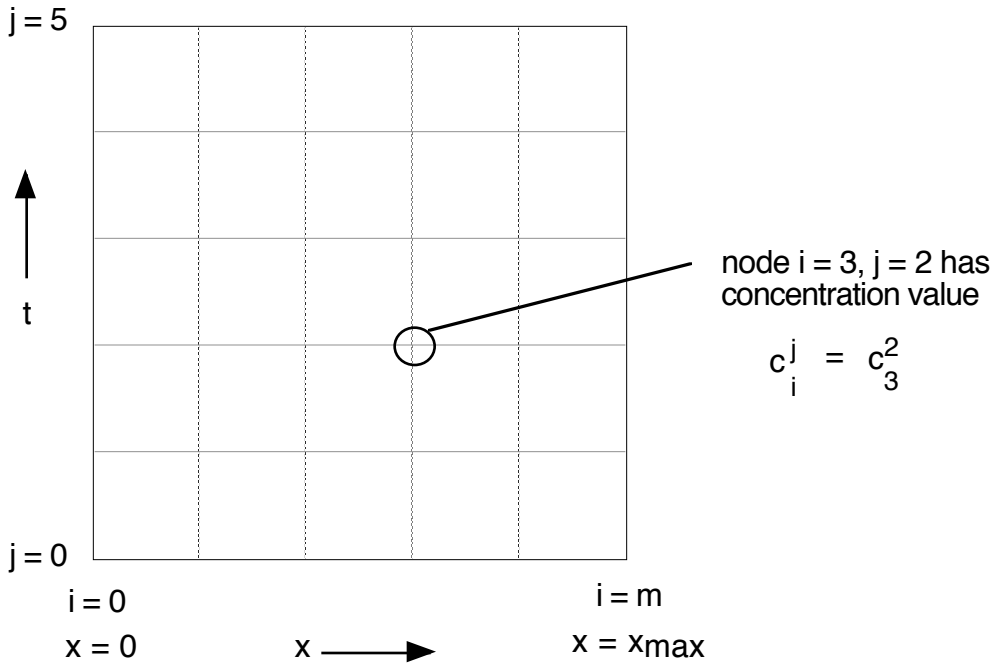
Note that this mass diffusion equation has an analogous form for heat conduction, with temperature, T, replacing concentration, c, and thermal diffusivity, $\alpha = k_t / \rho C_p$, replacing the mass diffusivity D.

Since there is variation with time we need an initial condition and since there is a second derivative with respect to position we need two boundary conditions.

Initial Condition: $c(x,0) = K_1$
Boundary condition at $x = 0$, $c(0,t) = K_2$
Boundary condition at $x = x_{max}$

$$\left.\frac{\partial c}{\partial x}\right|_{x = x_{max}} = 0$$

Imagine that spatial position x and time t are represented by a grid, with each node of the grid representing a point in x and a point in time. The spacing of the nodes along the x direction is Δx and the spacing of the nodes along the t direction is Δt. We use the subscript i to indicate which x location we are at and the superscript j to indicate which t location we are at on the grid.



The first derivative of c with respect to x is approximated by the “finite difference” approximation

$$\frac{\partial c}{\partial x} \approx \frac{c_{i+1} - c_i}{\Delta x}$$

The second derivative of c with respect to x is approximated by the following “centered finite difference derivative”:

$$\frac{\partial^2 c}{\partial x^2} \approx \lim_{\Delta x \to 0} \left[\frac{\left.\frac{\partial c}{\partial x}\right|_{i+1} - \left.\frac{\partial c}{\partial x}\right|_{i}}{\Delta x} \right] \approx \frac{\frac{c_{i+1} - c_i}{\Delta x} - \frac{c_i - c_{i-1}}{\Delta x}}{\Delta x}$$

$$\frac{\partial^2 c}{\partial x^2} \approx \frac{c_{i+1} - 2\,c_i + c_{i-1}}{(\Delta x)^2}$$

The time derivative is approximated by the following “forward finite divided difference” which is equivalent to using Euler’s method to integrate in time:

$$\frac{\partial c}{\partial t} \approx \frac{c_i^{j+1} - c_i^{j}}{\Delta t}$$

The finite difference approximation to our PDE is:

$$\frac{c_i^{j+1} - c_i^{j}}{\Delta t} = D\,\frac{c_{i+1}^{j} - 2\,c_i^{j} + c_{i-1}^{j}}{(\Delta x)^2}$$

Rearranging, we have the result for the concentration c at the next time step, j+1, as a function of concentrations at the current time step j:

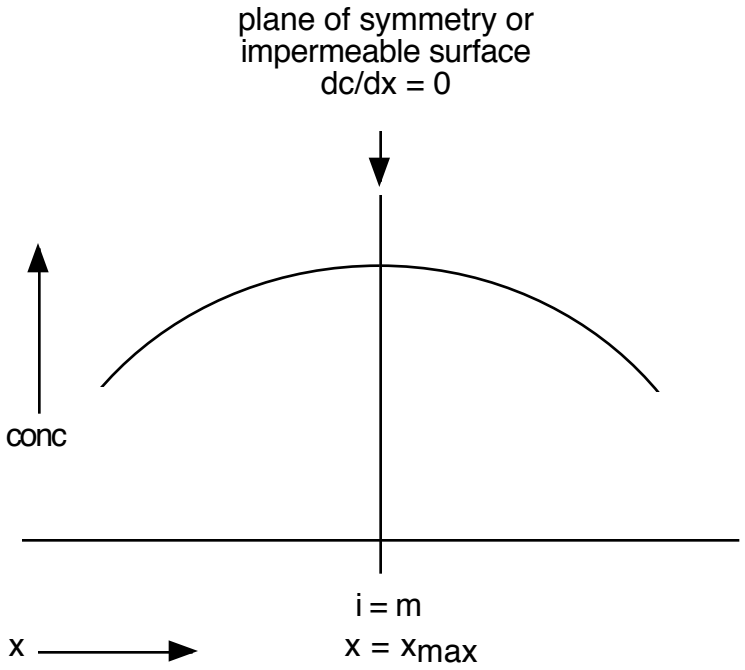
$$c_i^{j+1} = c_i^{j} + \lambda \left(c_{i+1}^{j} - 2\,c_i^{j} + c_{i-1}^{j} \right)$$

$$\lambda = \frac{D\,\Delta t}{(\Delta x)^2}$$

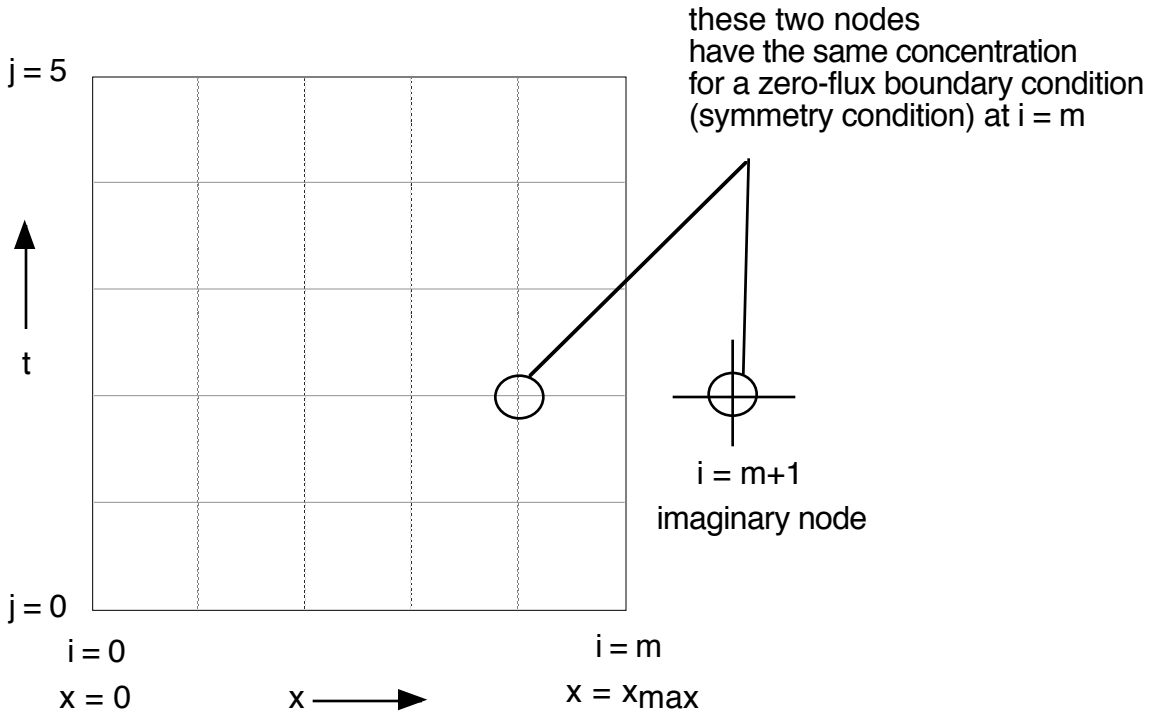
The above equation can be applied at all the internal nodes, that is, all nodes except those at the boundaries: node i = 0 representing position x = 0 and node i = m representing $x = x_{max}$.

At x = 0, the boundary condition above specifies that c_0 equals the constant K_2.

At $x = x_{max}$, the boundary condition above is the “zero flux” boundary condition. This type of boundary condition is also encountered at planes of symmetry:



We can formulate the difference equation for the node at i = m, $x = x_{max}$, by referencing an imaginary node at an imaginary grid location i = m + 1, where this imaginary node has the same concentration c as the node at i = m – 1. This would be the case for a concentration profile that is symmetrical about the plane at $x = x_{max}$, i = m.



Modifying the difference approximation for an internal node so that it applies to the node at i = m, we get:

$$c_m^{j+1} = c_m^j + \lambda \left(c_{m+1}^j - 2\, c_m^j + c_{m-1}^j \right)$$

$$c_m^{j+1} = c_m^j + \lambda \left(2c_{m-1}^j - 2\, c_m^j \right)$$

$$\lambda = \frac{D\, \Delta t}{(\Delta x)^2}$$

Stability and convergence criterion

In order to obtain a stable, non oscillatory and converging solution, the grid spacings Δt and Δx must be selected in order to meet the following criterion:

$$\lambda \leq 0.25$$

You will probably need to have more nodes - a finer grid spacing - than shown in the illustrations above.