Homework Assignment Number Six
Assigned: Tuesday, November 9, 1999
Due: Tuesday, November 23, 1999 BEGINNING OF CLASS.

In the file PDE505.zip, there is a code syspde_para.m with input file syspde_para_input.m that will solve systems of linear or nonlinear parabolic PDEs. Use this code to solve the following problems.

Problem (1) Combined diffusion with exothermic reaction problem (transient analysis)

Consider a cylindrical rod of nanoporous solid material. Inside this rod a reactant A is being converted to product B in a first order, irreversible reaction.

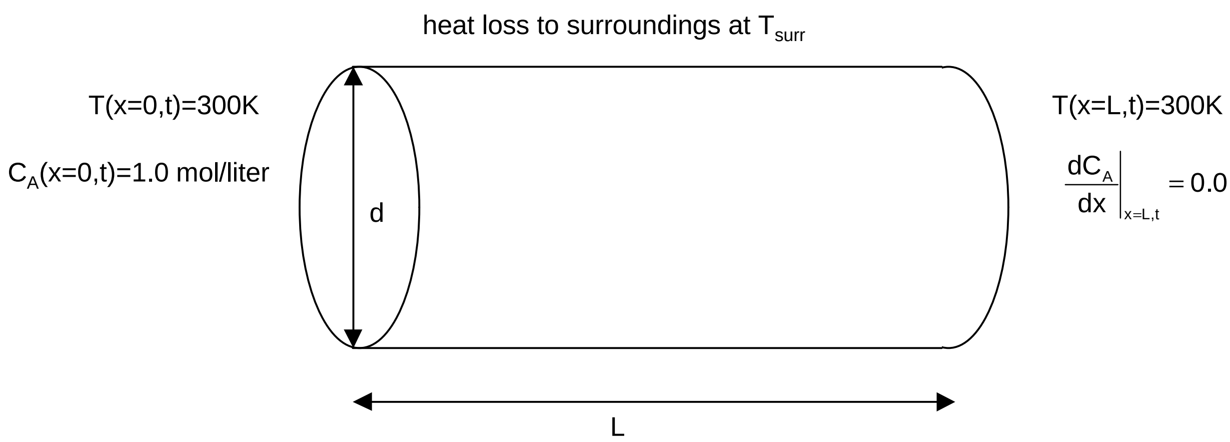


Figure One. Schematic of System

L is the length of the rod and d is the diameter.

If we assume that the distribution of A is uniform in the radial dimension, then a molar balance on component A yields:

acc = in - out + gen

$$\frac{\partial C_A}{\partial t} = D\frac{\partial^2 C_A}{\partial x^2} - kC_A \qquad (1)$$

where

C_A is the concentration of A [moles/liter]
D is the diffusivity of A in the rod [m^2/sec]
t is time [sec]
x is spatial position [m]

The rate constant, k, is given by

$$k = k_o e^{\frac{-E_a}{RT}} \qquad (2)$$

where

k_o is the exponential prefactor [1/sec]

E_a is the activation energy [Joules/mole]
T is the temperature [K]
R is the gas constant [8.314 J/mole/K]

Heat is lost to the surroundings at the surface of the cylinder. Heat is also generated via the exothermic reaction. Therefore an energy balance can be written as

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2} - \frac{\Delta H_r k C_A}{\rho C_p} + \frac{h A_{surf}}{\rho C_p V}(T_{surr} - T) \qquad (3)$$

where

$$\alpha = \frac{k_c}{\rho C_p}$$

and where

α is the thermal diffusivity [m^2/sec]
ΔH_r is the heat of reaction [J/mol]
ρ is the fluid molar density [mol/liter]
C_p is the fluid heat capacity [J/mol/K]
k_c is the thermal conductivity of the medium[J/m/K/sec]
h is the heat transfer coefficient [J/m^2/K/sec]
A_{surf} is the surface area of the rod [m^2]
V is the volume of the system [m^3]
T_{surr} is the temperature of the surroundings [K]

Initially, the entire nanoporous material contains A at a concentration of $C_{A,o}$ and at a temperature of T_o.

$$C_A(x, t = 0) = C_{A,o} \qquad (4.a)$$

$$T(x, t = 0) = T_o \qquad (4.b)$$

The temperature and the concentrations at x=0 are maintained at constant values.

$$C_A(x = 0, t) = C_{A,1} \qquad (5.a)$$

$$T(x = 0, t) = T_1 \qquad (5.b)$$

At the other end of the rod, the flux of A is zero and the temperature is maintained at a constant value.

$$\left.\frac{dC_A}{dx}\right|_{c=L,t} = 0.0 \qquad (5.c)$$

$$T(x = L, t) = T_2 \qquad (5.d)$$

Thus we have a fully specified system of coupled nonlinear parabolic partial differential equations in one spatial dimension.

Your task is to find the steady state and transient behavior of the Temperature and Concentration of A. You are to submit:

(a) your input file, containing the PDEs. (Not the entire syspde_para.m code.)

(b) The reduced temperature and concentration profiles (T/T_o & $C_A/C_{A,o}$) at time = 10, 100, 1000, 4000, & 16000 seconds. (5 plots total.)

(c) Discussion, including qualitative explanation of the behavior. Which term (diffusion, reaction, or heat loss) dominates in the energy balance? Why? Which term (diffusion or reaction) dominates in the mass balance? Why? Which term is causing the behavior seen in the profiles at time = 100 sec? Why? Which term is causing the behavior seen in the profiles at time = 16000 sec? Why?

parameters:

$C_{A,o} = 1.0$	[moles/liter]
$C_{A,1} = 1.0$	[moles/liter]
$T_o = 400$	[K]
$T_1 = 400$	[K]
$T_2 = 400$	[K]
$D = 1.0 \cdot 10^{-7}$	[m^2/sec]
$L = 0.2$	[m]
$d = 0.01$	[m]
$k_o = 5000.0$	[1/sec]
$E_a = 4 \cdot 10^4$	[Joules/mole]
$R = 8.314$	[J/mole/K]
$\Delta H_r = -5 \cdot 10^6$	[J/mol]
$\rho = 100.0$	[kg/m^3]
$C_p = 4000.0$	[J/kg/K]
$k_c = 200.0$	[J/m/K/sec]
$h = 4.0$	[J/m^2/K/sec]
$A_{surf} = \pi d L$	[m^2]
$V = \pi \frac{d^2}{4} L$	[m^3]
$T_{surr} = 250$	[K]

Solution:

I used the code syspde_para.m
The boundary condition and initial conditions from that function are shown below:

```
function y_out = syspde_boundary_conditions(k_eq,k_bc,x,t,y1,i_x, dx);
if (k_bc == 1)
    if (k_eq == 1)
    y_out = 1.0;
    elseif (k_eq == 2)
         y_out = 400;
   end
else
   if (k_eq == 1)
         y_out = y1(k_eq, i_x-2)- 2*dx*(0.0);
    elseif (k_eq == 2)
         y_out = 400;
   end
end

function y_out = syspde_initial_conditions(k_eq,x);
if (k_eq == 1)
   y_out = 1.0;
elseif (k_eq == 2)
    y_out = 400;
end
```

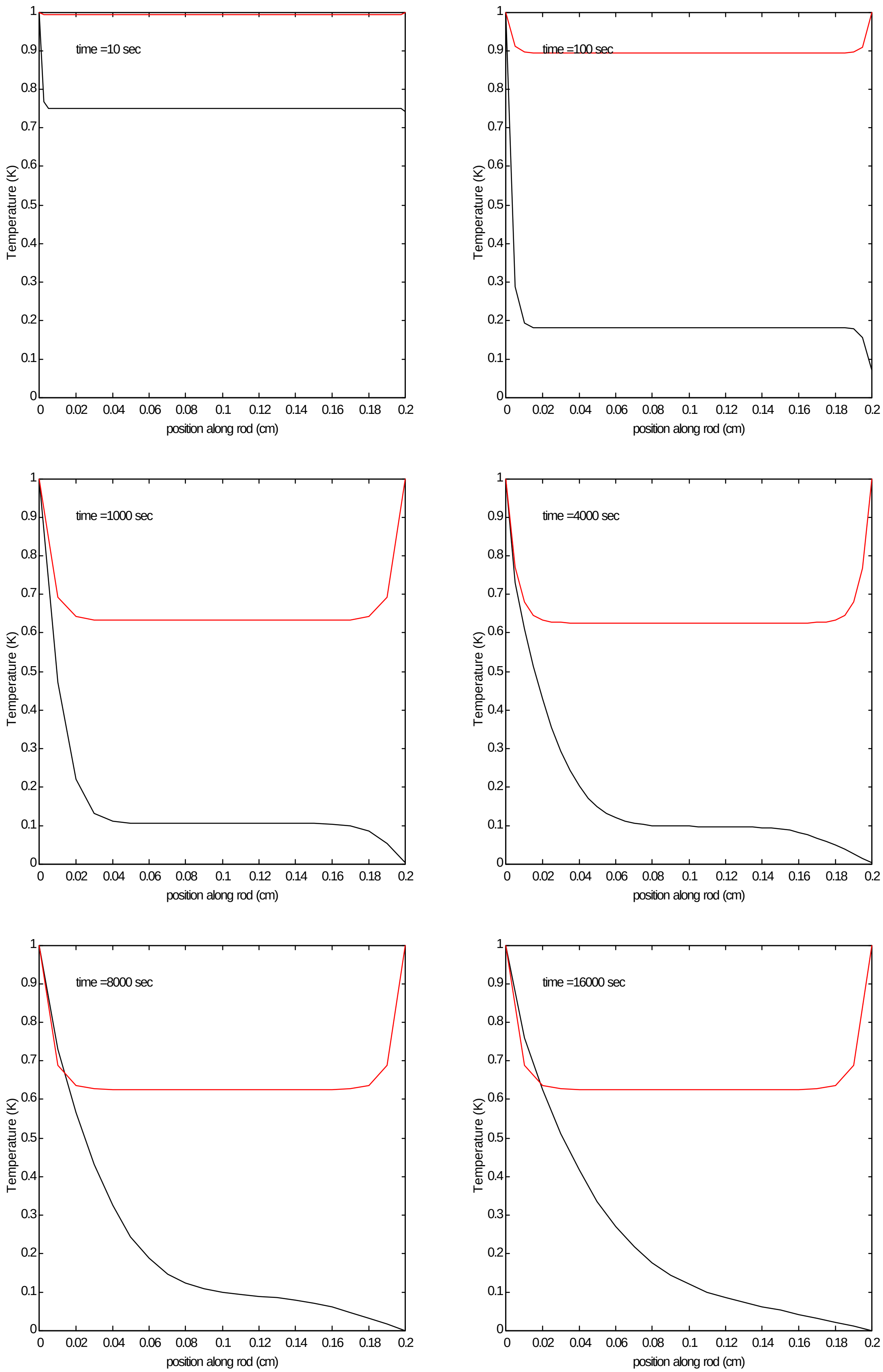
The code from the input file, syspde_para_input.m, is shown here in its entirety.

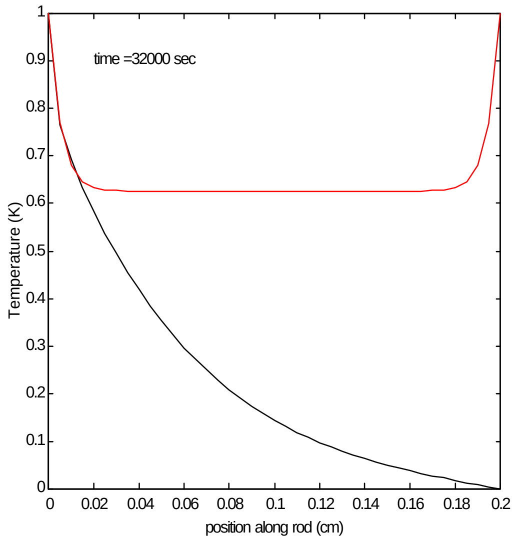
```
function dydt = syspde_para_input(n_eq,i_x_v,y1,t,dxi,mx)
% evaluate spatial derivatives
for k_eq = 1:1:n_eq
   for i_x = i_x_v(k_eq,1):1:i_x_v(k_eq,2)
    %dydx_matrix(k_eq, i_x) = 0.5*( y1(k_eq,i_x+1) - y1(k_eq,i_x-1) )*dxi;
      d2ydx2_matrix(k_eq,i_x) = ( y1(k_eq,i_x+1) - 2.0*y1(k_eq,i_x) + y1(k_eq,i_x-1) )*dxi^2;
   end
end

%
%  evaluate temporal derivative for all equations at interior nodes
%
dydt = zeros(n_eq,mx);
%  input parameters
L = 0.2;
d = 0.01;
D = 1.0e-7;
ko = 5000.0;
Ea = 4.0e+4;
R = 8.314;
DHr = -5000000.0;
rho = 100.0;
Cp = 4000.0;
kc = 200.0;
h = 4.0;
Asurf = pi*d*L;
V = pi*d^2*0.25*L;
Tsurr = 250.0;
afac = h*Asurf/(rho*Cp*V);
ffac = afac*Tsurr;
rfac = DHr/(rho*Cp);


% for k_eq = 1
%  mass balance
k_eq = 1;
for i_x = i_x_v(k_eq,1):1:i_x_v(k_eq,2)
   T = y1(2,i_x);
   Ca = y1(1,i_x);
   k = ko*exp(-Ea/(R*T));
```

```
   rate = k*Ca;
   dydt(k_eq,i_x) = D*d2ydx2_matrix(k_eq,i_x) - rate;
end
% for k_eq = 2
%  energy balance
k_eq = 2;
for i_x = i_x_v(k_eq,1):1:i_x_v(k_eq,2)
   T = y1(2,i_x);
   Ca = y1(1,i_x);
   k = ko*exp(-Ea/(R*T));
   rate = k*Ca;
   dydt(k_eq,i_x) = D*d2ydx2_matrix(k_eq,i_x) - rate*rfac + ffac - afac*T;
end
```

I used 80 spatial intervals with 100 temporal intervals to see the behavior up to time = 10 sec.
I used 40 spatial intervals with 500 temporal intervals to see the behavior up to time = 100 sec.
I used 20 spatial intervals with 2000 temporal intervals to see the behavior up to time = 1000,4000,8000, & 16000 sec.

Sample plots follow.


time =10 sec
time =100 sec
time =1000 sec
time =4000 sec
time =8000 sec
time =16000 sec
Temperature (K)
position along rod (cm)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.02
0.04
0.06
0.08
0.12
0.14
0.16
0.18


time =32000 sec
Temperature (K)
position along rod (cm)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.02
0.04
0.06
0.08
0.12
0.14
0.16
0.18

Problem (2) Solve a single hyperbolic PDE.

$$\frac{\partial^2 U}{\partial x^2} = \frac{1}{c^2}\frac{\partial^2 U}{\partial t^2}$$

where $c = 2$.

Use the following initial and boundary conditions.

$$U(x = 0, t) = 0.0$$
$$U(x = L, t) = 0.0$$
$$U(x, t = 0) = a\left(x^2 - Lx\right) + b \cdot \sin\left(\frac{6\pi x}{L}\right)$$
$$\frac{dU}{dt}(x, t = 0) = 0.0$$

where L = 2.0 and a = 0.01 and b= 0.001. I was able to solve this problem from time = 0 to 2.5 seconds, using 40 spatial intervals and 1000 temporal intervals.
Turn in a plot of the position and velocity for t = 0, 1, and 2.5 seconds.

Solution:

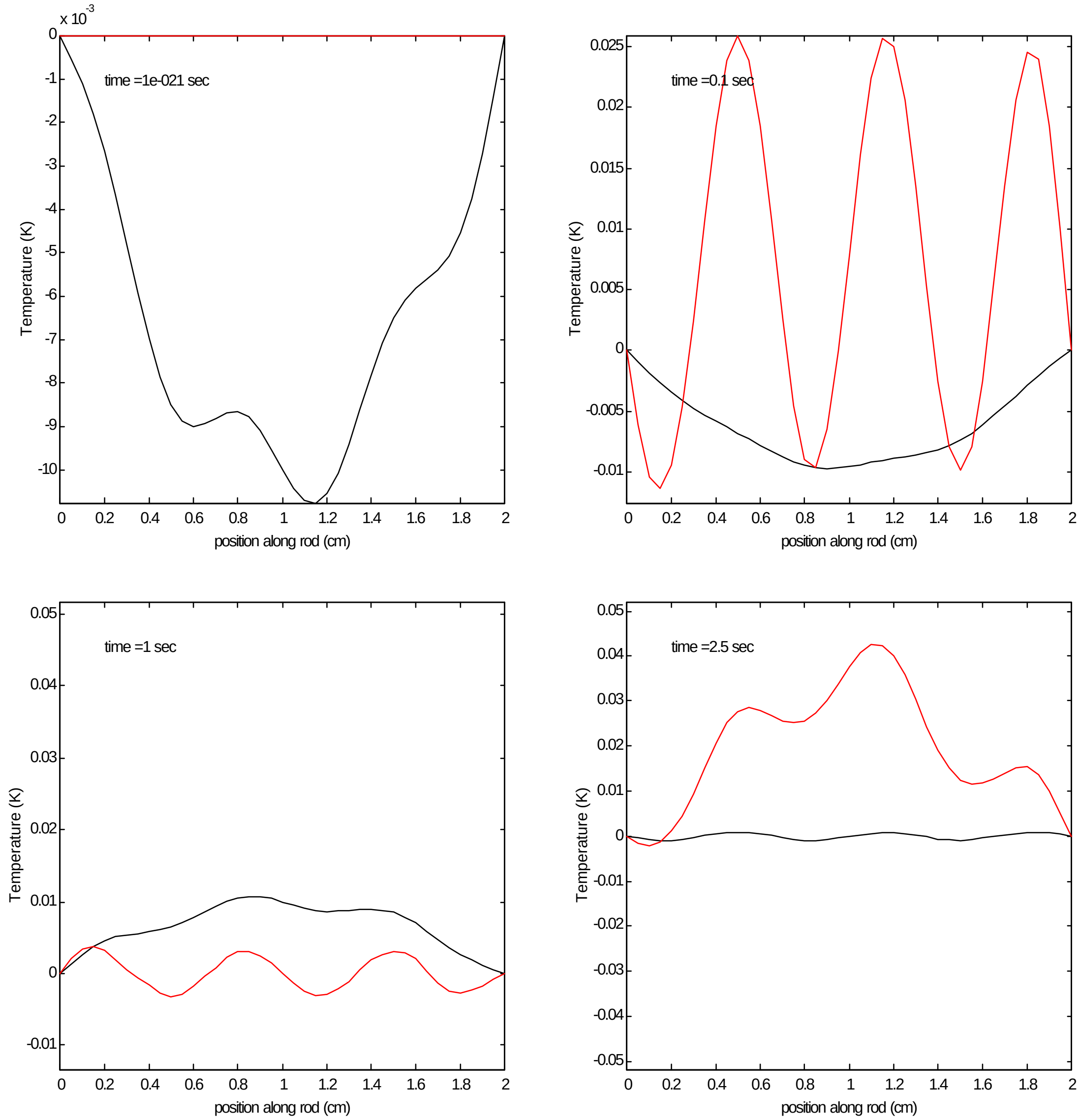
These were the initial and boundary condition functions:

```
function y_out = syspde_boundary_conditions(k_eq,k_bc,x,t,y1,i_x, dx);
if (k_bc == 1)
    if (k_eq == 1)
    y_out = 0;
    elseif (k_eq == 2)
        y_out = 0;
   end
else
    if (k_eq == 1)
    y_out = 0;
    elseif (k_eq == 2)
        y_out = 0;
   end
end
%fprintf(1,'BC: k_eq= %i kbc= %i i_x = %i x = %f t = %f yout = %f \n',k_eq, k_bc,i_x,x,t,y_out)


%
%  INITIAL CONDITION FUNCTION
%
function y_out = syspde_initial_conditions(k_eq,x);
global xo xf
if (k_eq == 1)
   a = 0.01;
   b = 0.001;
   L = xf - xo;
   %y_out = a*(x^2-L*x);
   y_out = a*(x^2-L*x) + b*sin(6*pi*x/L);
elseif (k_eq == 2)
    y_out = 0;
end
```

here is the PDE function from syspde_para_input.m

```
function dydt = syspde_para_input(n_eq,i_x_v,y1,t,dxi,mx)

% evaluate  spatial derivatives
%for k_eq = 1:1:n_eq
for k_eq = 1:1:1
   for i_x = i_x_v(k_eq,1):1:i_x_v(k_eq,2)
    %dydx_matrix(k_eq, i_x) = 0.5*( y1(k_eq,i_x+1) - y1(k_eq,i_x-1) )*dxi;
      d2ydx2_matrix(k_eq,i_x) = ( y1(k_eq,i_x+1) - 2.0*y1(k_eq,i_x) + y1(k_eq,i_x-1) )*dxi^2;
   end
end
%
c = 2;
c2 = c^2;
%
%  evaluate temporal derivative for all equations at interior nodes
%
dydt = zeros(n_eq,mx);
% for k_eq = 1
%  position
k_eq = 1;
   for i_x = i_x_v(k_eq,1):1:i_x_v(k_eq,2)
      dydt(k_eq,i_x) = y1(2,i_x);
   end
% for k_eq = 2
%  velocity
k_eq = 2;
   for i_x = i_x_v(k_eq,1):1:i_x_v(k_eq,2)
      dydt(k_eq,i_x) = c2*d2ydx2_matrix(1,i_x);
   end
```

here are the sample plots of the solution (position is black and velocity is red):



Problem (3) Solve a single elliptic PDE.

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = f(x,y) = 0$$

Use the following boundary conditions.

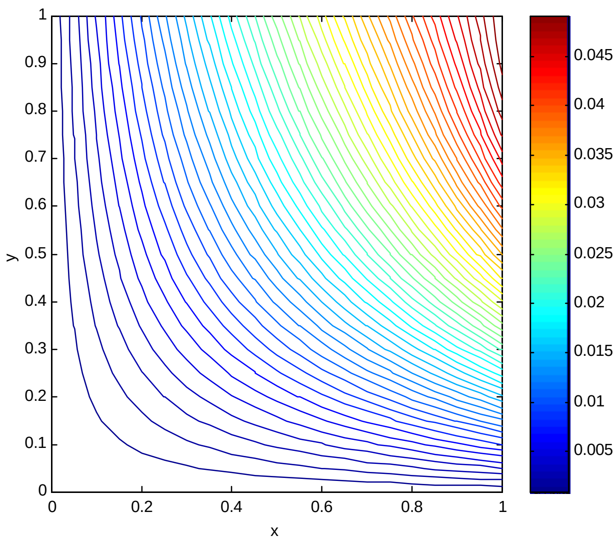
$$U(x=0,y) = 0.0$$
$$U(x=1,y) = \frac{y}{20}$$
$$U(x,y=0) = 0.0$$
$$U(x,y=1) = \frac{1}{20}\sin\left(\frac{\pi x}{2}\right)$$

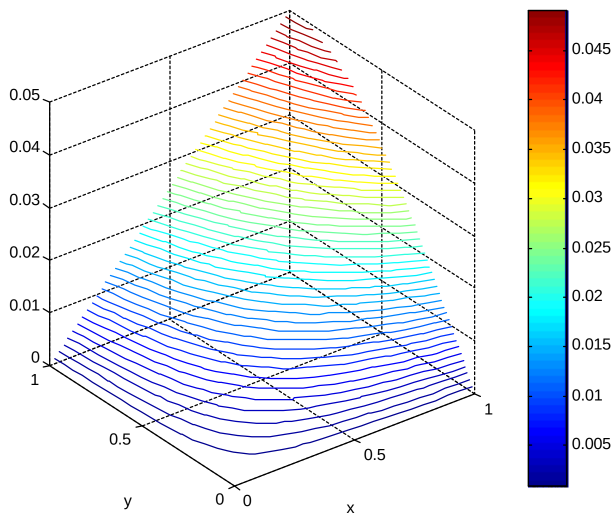
Show the U profile.

Solution:

I was able to solve the problem using Liebmann's method. I used lambda = 1.5. I used 20 spatial intervals in the x and y directions. My initial guess for the value of the interior nodes were all 0.0. It took 95 iterations to converge to a root mean square error less than 1.0e-6.

Here are the 2-D and 3-D color contour plots of the solution.





I used the MATLAB code from the notes on Liebmann's method in the elliptic PDE section of the course lecture material. I modified only the number of intervals, the value of lambda, the boundary conditions and the initial guesses.

Here is the code precisely as I used it.

```
function U = ell_liebmann
clear all
%
%  Solve Laplace's equation using Liebmann's method
%
%  Assume Dirichlet Boundary Conditions
%
%  Author:  David Keffer
%  Department of Chemical Engineering
%  University of Tennessee, Knoxville
%  Last Updated:  October 24, 2000
%

%solve for several values of lambda
%lambdav = [1.0:0.1:1.9];
lambdav = [1.5];
for k = 1:1:length(lambdav)
    lambda = lambdav(k);
     % number of intervals
     nx_int = 20;
     ny_int = 20;
     % number of nodes
     nx = nx_int + 1;
     ny = ny_int + 1;
     %  grid points in x and y
     xo = 0.0;
     xf = 1.0;
     yo = 0.0;
     yf = 1.0;
     dx = (xf - xo)/nx_int;
     dy = (yf - yo)/ny_int;
     xgrid = [xo:dx:xf];
     ygrid = [yo:dy:yf];
     % initialize U and Uold
     U = zeros(nx,ny);
     Uold = zeros(nx,ny);
     %  Fill in Uold with four dirichlet BCs
     for j = 1:1:ny
```

```
    % BC for x = xo
        Uold(1,j) = 0.0;
    % BC for x = xf
    Uold(nx,j) = ygrid(j)/20;
    % BC for y = yo
        Uold(j,1) = 0.0;
    % BC for y = yf
    Uold(j,ny) = 0.05*sin(pi*xgrid(j)*0.5);
    end
    %  fill in the interior nodes of Uold with initial guess
    for i = 2:1:nx-1
    for j = 2:1:ny-1
        Uold(i,j) = 0.0;
       end
    end
    U = Uold;
    %
    %  iteratively solve
    %
    maxit = 100;
    err = 100;
    tol = 1.0e-6;
    icount = 0;
    while (icount < maxit & err > tol)
    icount = icount + 1;
    Uold = U;
    for i = 2:1:nx-1
         for j = 2:1:ny-1
            % apply Liebmann's method
           U(i,j) = (U(i+1,j) + U(i-1,j) + U(i,j+1) + U(i,j-1) )*0.25;
           U(i,j) = lambda*U(i,j) + (1-lambda)*Uold(i,j);
        end
      end
      % calculate error
    err = 0.0;
    for i = 2:1:nx-1
       for j = 2:1:ny-1
          err = err + (U(i,j) - Uold(i,j))^2;
       end
    end
    err = sqrt(err/((nx-2)*(ny-2)));
    end
    fprintf(1,' lamdba = %f iteration = %i , error = %f \n',lambda, icount, err);
    % plot
    xmin = xgrid(1);
    xmax = xgrid(nx);
    ymin = ygrid(1);
    ymax = ygrid(ny);
    ncontourlines = 50;
    Umin = min(min(U));
    Umax = max(max(U));
    if (abs(Umin-Umax) < 1.0e-8);
      fprintf(1,'Solution is a flat plane with value = %f \n',Umin)
   else
      plot_dimensions = 3;
      if (plot_dimensions == 3)
           contour3(xgrid, ygrid,U,ncontourlines);
        axis([xmin xmax ymin ymax Umin Umax])
      else
         contour(xgrid, ygrid,U,ncontourlines);
        axis([xmin xmax ymin ymax])
      end
      xlabel('x');
      ylabel('y');
      colorbar
    end
end
```