Homework No.3 (Differential equations refresher)
Solve the problem of a transient 1-D tubular reactor using:
a) Finite differences to discretize in space, and two different solvers (built-in functions) to solve in time via Matlab
b) Using Matlab built-in solver PDEPE.
c) FEATool software.
d) Compare results and use different plots to describe the performance.
e) Sensitivity analysis mist be part of your report.

1-D PDE

Tubular water treatment uv-light reactor

1-D : 1-direction, 1- Dimension
PDE: Partial differential equation
ODEs: Ordinary differential equations
CODEs: Coupled ordinary differential equations
AEs: Algebraic equations
NLAEs: nonlinear algebraic equations
BC: Boundary condition
IC: Initial condition

Problem: The equation of conservation of chemical species under a chemical reaction of decomposition can be represented with the PDE given below.

$$\frac{\partial C}{\partial t} = \underline{\nabla} \cdot \left(D \underline{\nabla} C\right) - \underline{v} \cdot \underline{\nabla} C - kC^n$$

If a tubular catalytic chemical reactor initially filled with an inert solvent (C=0) is fed by a stream of component "A" with a concentration of 1 kmol/m^3 (C=1) and speed of 1 m/s (v=1), calculate the distribution of "A" across the reactor and as a function of time $C(x,t)$. The dispersion coefficient of the component "A" is 0.02 m^2/s (D=0.02), the kinetic decomposition coefficient 0.05 s^{-1} (k=0.05). The chemical decomposition kinetics is first order (n=1)

The molar balance in axial direction for a 1-D flow can be written as:

$$\frac{\partial C}{\partial t} = D\frac{\partial^2 C}{\partial x^2} - v\frac{\partial C}{\partial x} - kC^n$$

The initial and boundary conditions are:

Initial condition:

$$C\Big|_{t=0} = 0 \qquad 0 \le x \le 1$$

Boundary conditions:

$$C\Big|_{x=0} = 1 \qquad t > 0$$

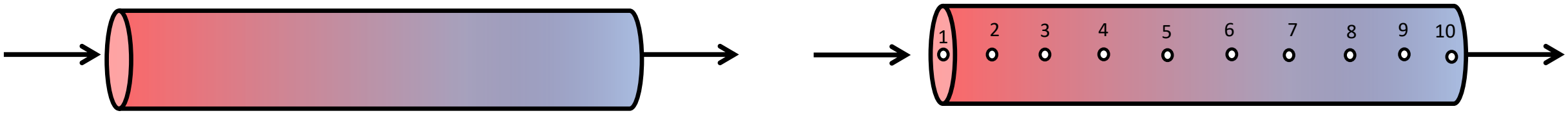
$$\frac{\partial C}{\partial x}\Big|_{x=L} = 0 \qquad t \ge 0$$

a) Use any integration technique (Runge-Kutta type scheme) to solve in time, and finite differences to integrate in axial domain.
b) Use the routine “pdepe” (finite element) to solve the 1-D PDE problem, and compare with the previous result.

By discretization, we refer to transform a PDF, which is continuous form in space (x) and time (t) into a set of ODEs, which are continuous in time and discrete in space. Mathematically speaking PDE in space and time transformed into a set of coupled ordinary differential equations

Discretization in the “x” domain

$$C(x,t) \rightarrow C_i(t) = C(x_i,t)$$



a) Use any integration technique (Runge-Kutta type scheme) to solve in time, and finite differences to integrate in axial domain.
b) Use the routine “pdepe” to solve the 1-D PDE problem, and compare with the previous result.

Using central finite differences in space (axial direction), the equation takes the form:

PDE form:

$$\frac{\partial C}{\partial t} = D\frac{\partial^2 C}{\partial x^2} - v\frac{\partial C}{\partial x} - kC^n$$

Central differences discretization $o(h^2)$

ODEs Form : (for i=2,3,4, …, N-1)

$$\frac{dC_i}{dt} = D\frac{C_{i+1} - 2C_i + C_{i-1}}{(\Delta x)^2} - v\frac{(C_{i+1} - C_{i-1})}{2\Delta x} - kC_i^n$$

BC Form:

$$C\Big|_{x=0} = 1 \qquad t > 0$$

$$\frac{\partial C}{\partial x}\Big|_{x=L} = 0 \qquad t \geq 0$$

Backward differences discretization $o(h^2)$

AE Form:

$$C_1 = 1$$

$$\frac{dC_N}{dx} = \frac{C_{N-2} - 4C_{N-1} + 3C_N}{2\Delta x} = 0$$

See Appendix for approximation of derivatives

Using RK45 and Finite diffrences

Main program (this code was written for 10 nodes or 8 internal nodes)

Parameters

Domain of time

Initial conditions of the resulting set of ODEs

Integration scheme

Plot results

```
% reactor by jose lopez salinas
 p(1)=0.01;    %Diffusion coefficient D
 p(2)=1;       % c0, injection concentrtion
 p(3)=0.5;     % k , first order kinetic coefficient
 p(4)=1;       % vo , velocity of fluid injection
 Tspan=[0 1];
 Y0(1)=1.0;Y0(2)=0.0;Y0(3)=0.0;Y0(4)=0.0;Y0(5)=0.0;
 Y0(6)=0.0;Y0(7)=0.0;Y0(8)=0.0;Y0(9)=0.0;
 OPTIONS=[];
 [time,Y]=ode45(@reactub,Tspan,Y0,[],p);
 plot(time,Y)
```

Function to define the ODEs

```
% tubular reactor by jose lopez salinas
function yprime=reactub(t,y,p)
 c=y;
 D=p(1);
 k=p(3);
 vo=p(4);
 m=1;
 yprime(1)=0; c(1)=1;
 dx=1/9.0;
 c(10)=(4*c(9)-c(8))/3;
 for i=2:9
    yprime(i)=D*(c(i+1)-2*c(i)+c(i-1))/(dx^2)-vo*(c(i+1)-c(i-1))/(2*dx)-k*c(i)^m;
 end
yprime=yprime';
```

$$C_1 = 1$$

$$\frac{dC_N}{dx} = \frac{C_{N-2} - 4C_{N-1} + 3C_N}{2\Delta x} = 0$$

$$\frac{dC_i}{dt} = D\frac{C_{i+1} - 2C_i + C_{i-1}}{(\Delta x)^2} - v\frac{(C_{i+1} - C_{i-1})}{2\Delta x} - kC_i^n$$

The problem can be solved as ODEs (representing the PDE), and algebraic equations (representing the boundary conditions) as shown above, but you may also decide to put down everything in terms of ODEs, for instance

$$C_1 = 1 \quad \frac{dC_1}{dt} = 0$$

$$\frac{dC_N}{dx} = \frac{C_{N-2} - 4C_{N-1} + 3C_N}{2\Delta x} = 0 \quad \frac{dC_{10}}{dt} = \frac{1}{3}\left(4\frac{dC_9}{dt} - \frac{dC_8}{dt}\right)$$

Same as the previous code, but generalized for N nodes

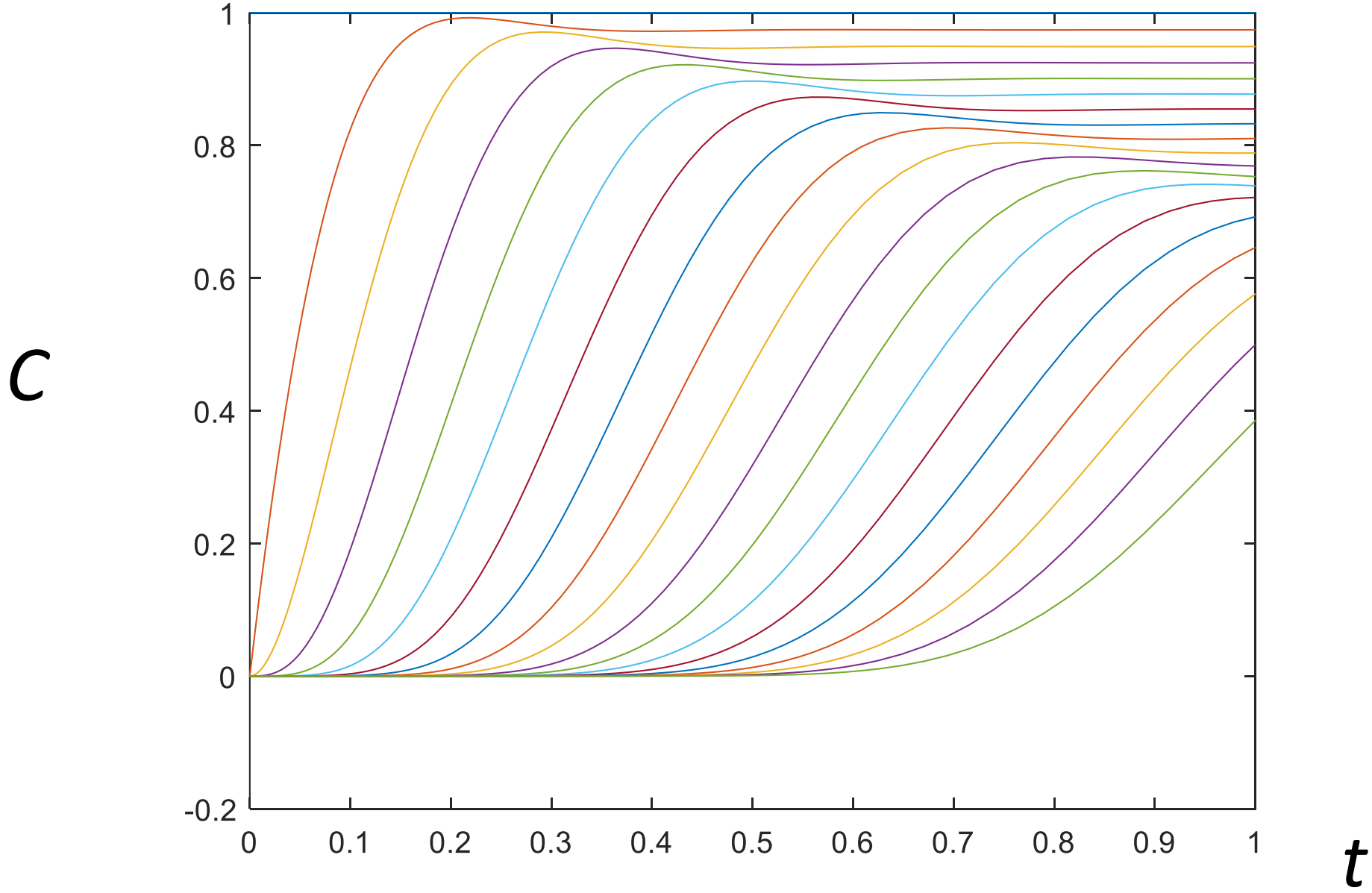
Main program (this code was written for N nodes)

```
 p(1) = 0.01;    %Diffusion coefficient D
 p(2) = 1;       % c0
 p(3)=0.5;       % k , first order kinetic coefficient
 p(4)=1;         % vo , velocity of fluid injection
 M=20;
 p(5)=M;
Tspan=[0 1];
Y0(1)=1.0;
for i=2:M
    Y0(i)=0;
end
OPTIONS=[];
[time,Y]=ode45(@reactub,Tspan,Y0,[],p);
plot(time,Y)
```

Same as the previous code, but generalized for N nodes, and without algebraic equations

Function to define the ODEs without the algebraic equations

```
% tubular reactor by jose lopez salinas
function yprime=reactub(t,y,p)
c=y;
 D = p(1);
 k=p(3);
 vo=p(4);
 N=p(5);
 m=1;
yprime(1)=0;
dx=1/(N-1);
for i=2:N-1
    yprime(i)=D*(c(i+1)-2*c(i)+c(i-1))/(dx^2)-vo*(c(i+1)-c(i-1))/(2*dx)-k*c(i)^m;
End
yprime(N)=(4*yprime(N-1)-yprime(N-2))/3;
yprime=yprime';
```


c
1
0.8
0.6
0.4
0.2
0
-0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
t

$u = u\,(x,t)$

1D-PDE

Finite Element

The built in function PDEPE, solves a general problem of a 1-D (parabolic or elliptic) partial differential equation, for a Cartesian, cylindrical or spherical coordinates of the from:

PDE

$$c\left(x,t,u,\frac{\partial u}{\partial x}\right)\frac{\partial u}{\partial t} = \frac{1}{x^m}\frac{\partial}{\partial x}\left(x^m f\left(x,t,u,\frac{\partial u}{\partial x}\right)\right) + s\left(x,t,u,\frac{\partial u}{\partial x}\right)$$

$$c\frac{\partial u}{\partial t} = \frac{1}{x^m}\frac{\partial}{\partial x}(x^m f) + s$$

The built in function PDEPE must contain any of the boundary conditions of the forms :

PDE- BC

Left boundary

$$p_L(x_L,t,u_L) + q_L(x_L,t)\left[f\left(x,t,u,\frac{\partial u}{\partial x}\right)\Big|_{x=x_L}\right] = 0$$

$$p_L + q_L f = 0$$

Right boundary

$$p_R(x_R,t,u_R) + q_R(x_R,t)\left[f\left(x,t,u,\frac{\partial u}{\partial x}\right)\Big|_{x=x_R}\right] = 0$$

$$p_R + q_R f = 0$$

The built in function PDEPE must have initial condition of the form:

$$u_o = g$$

PDE- IC

$$u(x,t=0) = u_o(x)$$

Matlab: pdepe (parabolic PDE)

m = 0 for slab, 1 cylinder, 2 sphere

How to translate the mathematical problem to the matlab function PDEPE:

Language of our problem

$$\frac{\partial C}{\partial t} = D\frac{\partial^2 C}{\partial x^2} - v\frac{\partial C}{\partial x} - kC^n$$

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} - v\frac{\partial u}{\partial x} - ku^n$$

Language using PDEPE

$$c\frac{\partial u}{\partial t} = \frac{1}{x^m}\frac{\partial}{\partial x}(x^m f) + s$$

$$c\frac{\partial u}{\partial t} = \frac{1}{x^m}\frac{\partial}{\partial x}\left(x^m D\frac{\partial u}{\partial x}\right) + \left[-v\frac{\partial u}{\partial x} - ku^n\right]$$

External functions not known by Matlab
User defined functions (needed)

$$\begin{cases} c = 1 \\ m = 0 \\ f = D\dfrac{\partial u}{\partial x} \\ s = \left[-v\dfrac{\partial u}{\partial x} - ku^n\right] \end{cases}$$

Language of our problem

Language using PDEPE

$$f = D\frac{\partial u}{\partial x}$$

BC Form:

$$C\Big|_{x=0} = 1 \qquad t > 0$$

$$p_L + q_L\,[f] = 0$$

$$\left.\frac{\partial C}{\partial x}\right|_{x=L} = 0 \qquad t \geq 0$$

$$p_R + q_R[f] = 0$$

$$u\Big|_{x=0} = 1 \qquad t > 0$$

$$\begin{cases} \left(u\Big|_{x=0} - 1\right) + q_L\left[D\left.\frac{\partial u}{\partial x}\right|_{x=0}\right] = 0 \\ (u_L - 1) + 0\left[D\left.\frac{\partial u}{\partial x}\right|_{x=x_L}\right] = 0 \end{cases} \quad \begin{cases} p_L = (u_L - 1) \\ q_L = 0 \end{cases}$$

$$\left.\frac{\partial u}{\partial x}\right|_{x=L} = 0 \qquad t \geq 0$$

$$\begin{cases} p_R + q_R\left[D\left.\frac{\partial u}{\partial x}\right|_{x=0}\right] = 0 \\ 0 + q_R\left[D\left.\frac{\partial u}{\partial x}\right|_{x=0}\right] = 0 \end{cases} \quad \begin{cases} p_R = 0 \\ q_R = 1 \end{cases}$$

Using PDEPE

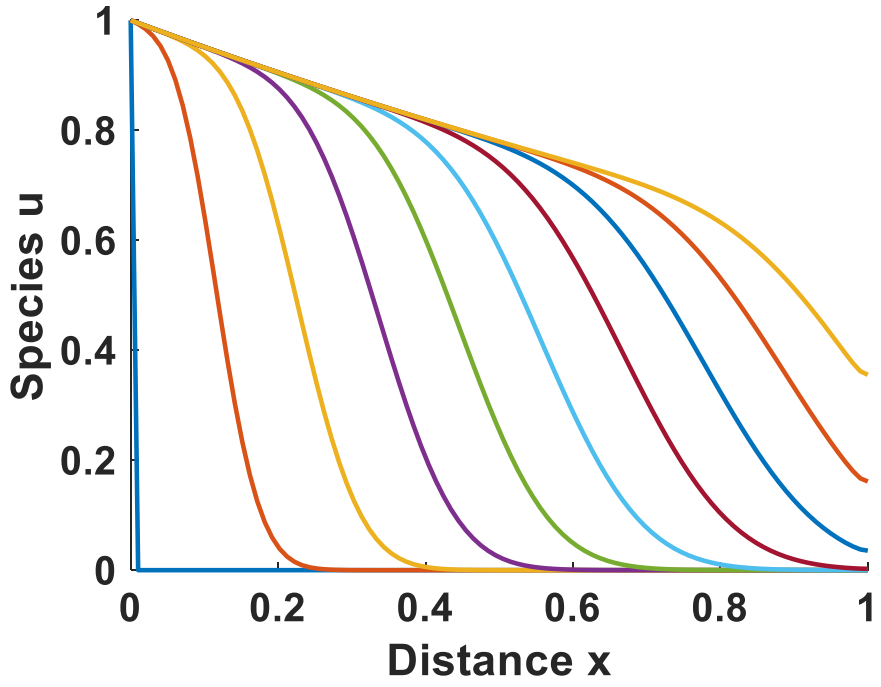
```
function [c,f,s] = DiffusionPDEfun(x,t,u,dudx,P)
% Function defining the PDE
% C du/dt =(1/x^n)d(x^n f)/dx+s
% f=f(x,t,u,du/dx)
% s=s(x,t,u,du/dx)
% c=c(x,t,u,du/dx)
% Extract parameters
D = P(1);
k=P(3);
vo=P(4);
% PDE
c = 1;
f = D.*dudx;
s =-k*u-vo*dudx
```

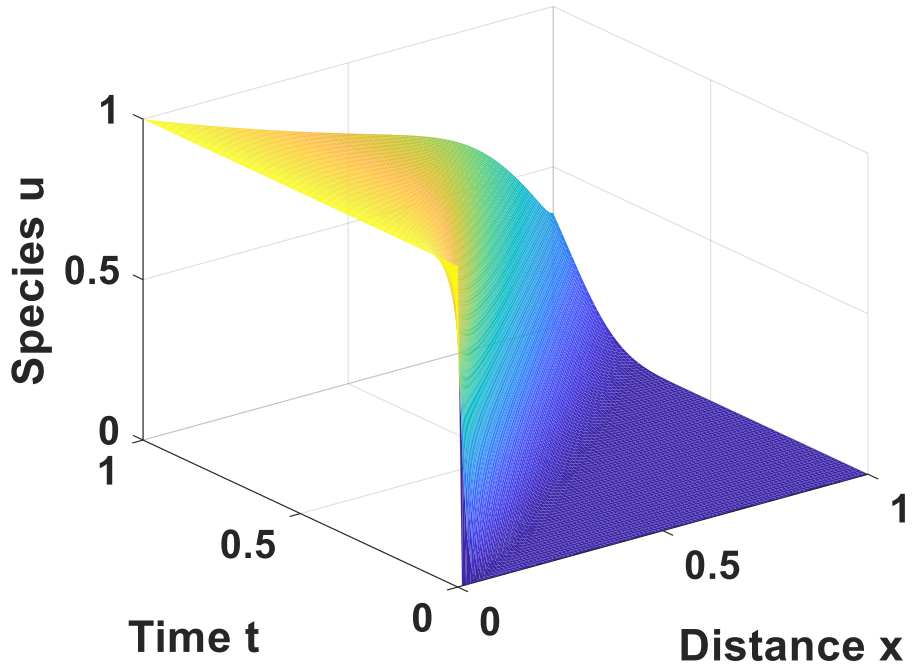
```
function [pl,ql,pr,qr] = DiffusionBCfun(xl,u1,xr,ur,t,P)
% Boundary conditions for x = 0 and x = L;
% pL(xl,t,ul)+qL(xl,t)*f(x,t,u,du/dx)=0
% pR(xr,t,ur)+qR(xr,t)*f(x,t,u,du/dx)=0
% Extract parameters
c0 = P(2);
% BCs: No flux boundary at the right boundary and constant concentration on
% the left boundary
pl = u1-c0; ql = 0; pr = 0; qr = 1;
```

```
function u0 = DiffusionICfun(x,P)
% Initial conditions for t = 0; can be a funciton of x
% u0=u0(x)
u0 = 0;
```

```
function[t,x,u]= Diffusion
 % This is the main function. Within this function the meshes are defined,
 % PDEPE is called and the results are plotted
 clear; close all;

 %% Parameters
 %% dC/dt=div(D Grad C)-k*C- dot(v,Grad C)
 %% dC/dt= D d(dC/dx)/dx- k*C-v*dC/dx
 %% in the previous pde, all are partial derivatives
 P(1) = 0.01;    %Diffusion coefficient D
 P(2) = 1;       % c0
 P(3)=0.5;       % k , first order kinetic coefficient
 P(4)=1;         % vo , velocity of fluid injection
 L = 1; %Length of domain
 maxt = 1; %Max. simulation time
 m = 0; %Parameter corresponding to the symmetry of the problem (see help)
 t = linspace(0,maxt,100); %tspan
 x = linspace(0,L,100); %xmesh

 %%
 % Call of PDEPE. It needs the following arguments
 % m: see above
 % DiffusionPDEfun: Function containg the PDEs
 % DiffusionICfun: Function containing the ICs for t = 0 at all x
 % DiffusionBCfun: Function containing the BCs for x = 0 and x = L
 % x: xmesh and t: tspan
 % PDEPE returns the solution as multidimensional array of size
 % xmesh x tspan x (# of variables)
 sol = pdepe(m,@DiffusionPDEfun,@DiffusionICfun,@DiffusionBCfun,x,t,[],P);
 u = sol
```

```
%% Plotting
% 3D surface plot

figure(1)
surf(x,t,u,'edgecolor','none');
xlabel('Distance x','fontsize',20,'fontweight','b','fontname','arial')
ylabel('Time t','fontsize',20,'fontweight','b','fontname','arial')
zlabel('Species u','fontsize',20,'fontweight','b','fontname','arial')
axis([0 L 0 maxt 0 P(2)])
set(gcf(), 'Renderer', 'painters')
set(gca,'FontSize',18,'fontweight','b','fontname','arial')

% 2D line plot
figure(2)
hold all
for n = linspace(1,length(t),10)
plot(x,sol(n,:),'LineWidth',2)

end
xlabel('Distance x','fontsize',20,'fontweight','b','fontname','arial')
ylabel('Species u','fontsize',20,'fontweight','b','fontname','arial')
axis([0 L 0 P(2)])
set(gca,'FontSize',18,'fontweight','b','fontname','arial')
```


Species u
1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1
Distance x


Species u
1
0.5
0
1
0.5
0
Time t
0
0.5
1
Distance x

Appendix (Derivatives, using Taylor series)

Central Differences $O(h^2)$

$$\begin{bmatrix} 2\,h\,f^{I} \\ h^2\,f^{II} \\ 2\,h^3\,f^{III} \\ h^4\,f^{IV} \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 & 1 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ -1 & 2 & 0 & -2 & 1 \\ 1 & -4 & 6 & -4 & 1 \end{bmatrix} \begin{bmatrix} f_{i-2} \\ f_{i-1} \\ f_i \\ f_{i+1} \\ f_{i+2} \end{bmatrix}$$

Backward differences $O(h^2)$

$$\begin{bmatrix} 2\,h\,f^{I} \\ h^2\,f^{II} \\ 2\,h^3\,f^{III} \\ h^4\,f^{IV} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & -4 & 3 \\ 0 & 0 & -1 & 4 & -5 & 2 \\ 0 & 3 & -14 & 24 & -18 & 5 \\ -2 & 11 & -24 & 26 & -14 & 3 \end{bmatrix} \begin{bmatrix} f_{i-5} \\ f_{i-4} \\ f_{i-3} \\ f_{i-2} \\ f_{i-1} \\ f_i \end{bmatrix}$$

For internal discretization points

$$\frac{dy_i}{dx} = \frac{y_{i+1} - y_{i-1}}{2\,\Delta x}$$

$$\frac{d^2y_i}{dx^2} = \frac{y_{i+1} - 2y_i + y_{i-1}}{(\Delta x)^2}$$

For downstream boundary points

$$\frac{dy_i}{dx} = \frac{y_{i-2} - 4y_{i-1} + 3y_i}{2\,\Delta x}$$

Nomenclature of this slide

$$f_i = f(x)$$

$$f_{i+n} = f(x + h\Delta x)$$

$$f_{i-n} = f(x - h\Delta x)$$

$$f^N = \frac{d^N f}{dx^N}$$

Forward differences $O(h^2)$

$$\begin{bmatrix} 2\,h\,f^{I} \\ h^2\,f^{II} \\ 2\,h^3\,f^{III} \\ h^4\,f^{IV} \end{bmatrix} = \begin{bmatrix} -3 & +4 & -1 & 0 & 0 & 0 \\ +2 & -5 & +4 & -1 & 0 & 0 \\ -5 & 18 & -24 & +14 & -3 & 0 \\ +3 & -14 & +26 & -24 & +11 & 11 \end{bmatrix} \begin{bmatrix} f_i \\ f_{i+1} \\ f_{i+2} \\ f_{i+3} \\ f_{i+4} \\ f_{i+5} \end{bmatrix}$$

For upstream boundary points

$$\frac{dy_i}{dx} = \frac{-y_{i+2} + 4y_{i+1} - 3y_i}{2\,\Delta x}$$