function [x, phi] = convection\_diffusion\_fem(NP, u, k, L, alpha)

% Inputs:

% NP - Number of points

% u - Velocity

% k - Diffusion coefficient

% L - Total length of the domain

% alpha - Upwinding factor (Petrov-Galerkin weighting)

% Preprocessing

NE = NP - 1; % Number of elements

h = L / NE; % Length of each element

x = linspace(0, L, NP); % Mesh points

% Initialize global stiffness matrix and force vector

K\_global = zeros(NP, NP);

F\_global = zeros(NP, 1);

% Connectivity matrix

intma = zeros(NE, 2);

for i = 1:NE

intma(i, :) = [i, i+1];

end

% Assembly of element matrices

for i = 1:NE

% Local-to-global mapping

node1 = intma(i, 1);

node2 = intma(i, 2);

% Element convection and diffusion matrix

emat\_convection = (u/2) \* [-1 1; -1 1];

emat\_diffusion = (k/h) \* [1 -1; -1 1];

emat\_pg = (alpha \* u/2) \* [-1 1; -1 1]; % Petrov-Galerkin part

% Total element matrix

emat = emat\_convection + emat\_diffusion + emat\_pg;

% Assemble into global matrix

K\_global([node1, node2], [node1, node2]) = ...

K\_global([node1, node2], [node1, node2]) + emat;

end

% Apply boundary conditions

K\_global(1, :) = 0; K\_global(1, 1) = 1; F\_global(1) = 1; % Dirichlet BC at x=0

K\_global(NP, :) = 0; K\_global(NP, NP) = 1; F\_global(NP) = 0; % Dirichlet BC at x=L

% Solve the system of equations

phi = K\_global \ F\_global;

% Postprocessing

plot(x, phi);

title('Solution of Convection-Diffusion Equation using FEM');

xlabel('Domain (x)');

ylabel('Solution (\phi)');

end