ASSIGNMENT 1

FINITE ELEMENT FOR FLUIDS

PRADEEP KUMAR BAL

ADVISOR: - Prof. Pablo Saez

Problem Statement:-

1D convection-diffusion equation with constant coefficients and Dirichlet boundary conditions:

$au_x - \gamma u_{xx} = s \qquad x\epsilon[0,1]; \qquad u_0 = 0 \; and \; u_1 = 1$

Given: s=0; $\gamma = 0.01$,

Goal:-

- To solve the given problem using Galerkin, SU, SUPG, GLS, and SGS methods with 10 linear elements and 10 quadratic elements respectively with the optimal stabilization parameter.
- To observe the effect of the stabilization parameter and to investigate improvement in obtained solutions for more number of elements (30 quadric elements have been considered this case).

Solution:-

Péclet number (**Pe**) is defined to be the ratio of the rate of advection of a physical quantity by the flow to the rate of diffusion of the same quantity driven by an appropriate gradient. For 10 linear elements h=0.1**; Pe=** $\frac{h*|\boldsymbol{a}|}{2*\gamma} = 5 > 1$

Galerkin Approximation:-

The corresponding weak form is:

$\int_0^L (w * a * u_x + w_x * \gamma * u_x)\, dx = 0$

In this case the stabilisation parameter $\tau = 0$. $\bar{\gamma} = 0$

For 10 linear elements node to node oscillations are observed as here Pe > 1 (Figure 1(a)) as convection is dominated over diffusion. With 10 quadratic elements a bit more improved solutions with significantly lower oscillations are observed (Figure 2(a)). It can be concluded that the Galerkin method lacks enough diffusion.

Stream-lined upwind method (SU method):

It is an optimal technique which produces the exact solution at the nodes of a uniform mesh of linear elements. The corresponding weak form is

$\int_0^L (w * a * u_x + w_x * (\gamma + \bar{\gamma}) * u_x)\, dx = 0$ With $\bar{\gamma} = \frac{\beta * a * h}{2}$;

$\tau = \frac{\beta * h}{2a}$; $\beta = \coth(Pe) - 1/Pe$

In the given problem the source term is given as zero (constant), so for linear elements exact nodal solutions are observed (Figure 1(b)). Also for quadratic elements for this case improved solutions have been observed (Figure 3 (b)). One major drawback of this method is its non-consistent formulation does not perform well for nor constant source terms.

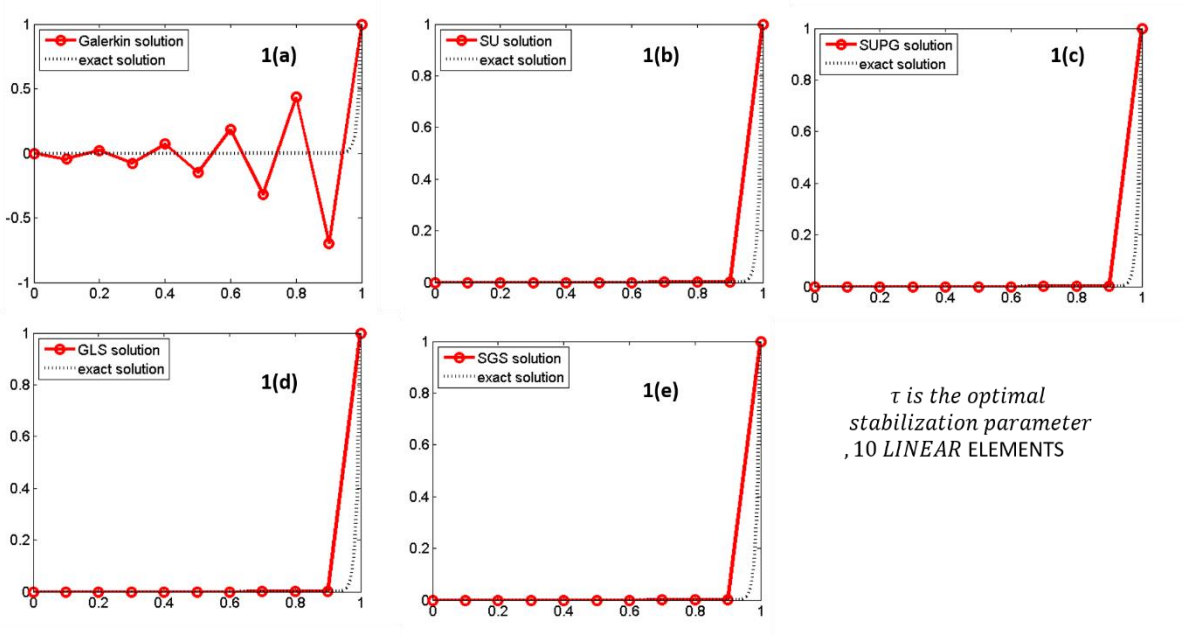

Figure 1: Comparison of solutions from different methods for 10 linear elements

SUPG Method:-

The stabilised consistent formulation for this method of the given problem is

a (w, u) +c(w,u, a)+$\sum_e \int_0^L (w_x * a * \tau * R(u))\, dx = 0$ where R(u)= $au_x - \gamma u_{xx}$

Here τ is considered as the optimal stabilisation parameter.

It is observed that for linear elements this method gives similar results as that of SU method as u_{xx} becomes zero (Figure 1 and 3). For quadratic elements this method gives more improved

results. With increasing number of elements the obtained solution tends to the exact solution. The added stabilization term is not symmetric. There are technical difficulties in establishing the stability of the GLS method.

One special case is studied for $\tau = 1$ with 10 linear elements for all the methods. The obtained results (Figure:-2) are comparatively less accurate than when τ is considered as the optimal.

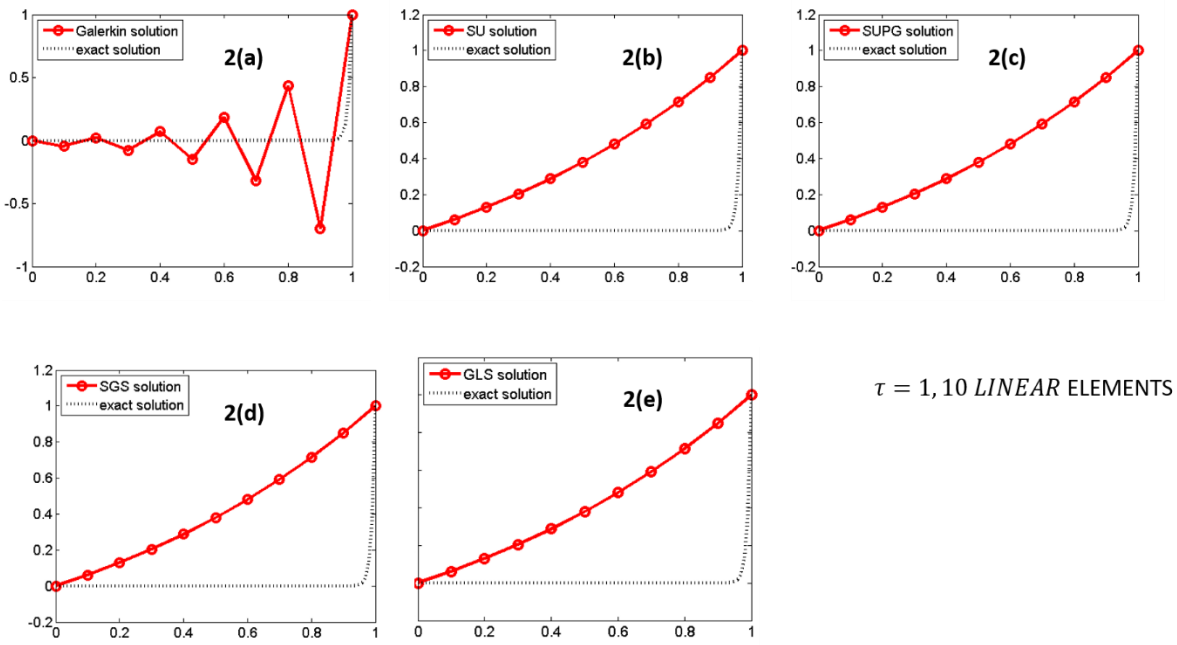

Figure 2: τ=1; 10 Linear elements

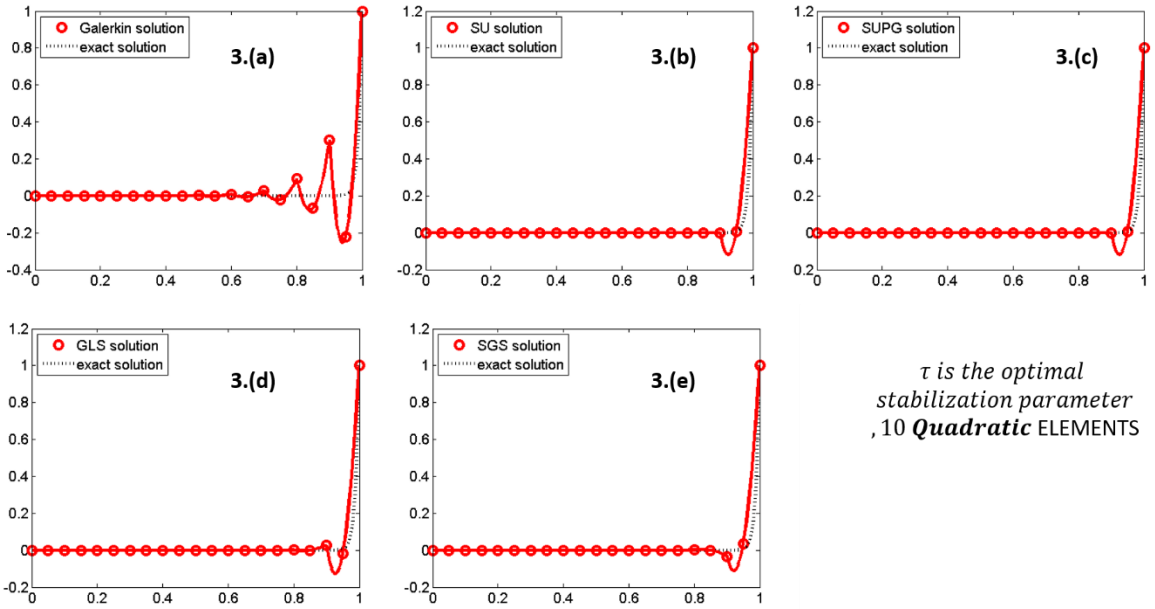

Figure 3: Comparison of solutions from different methods for 10 quadratic elements

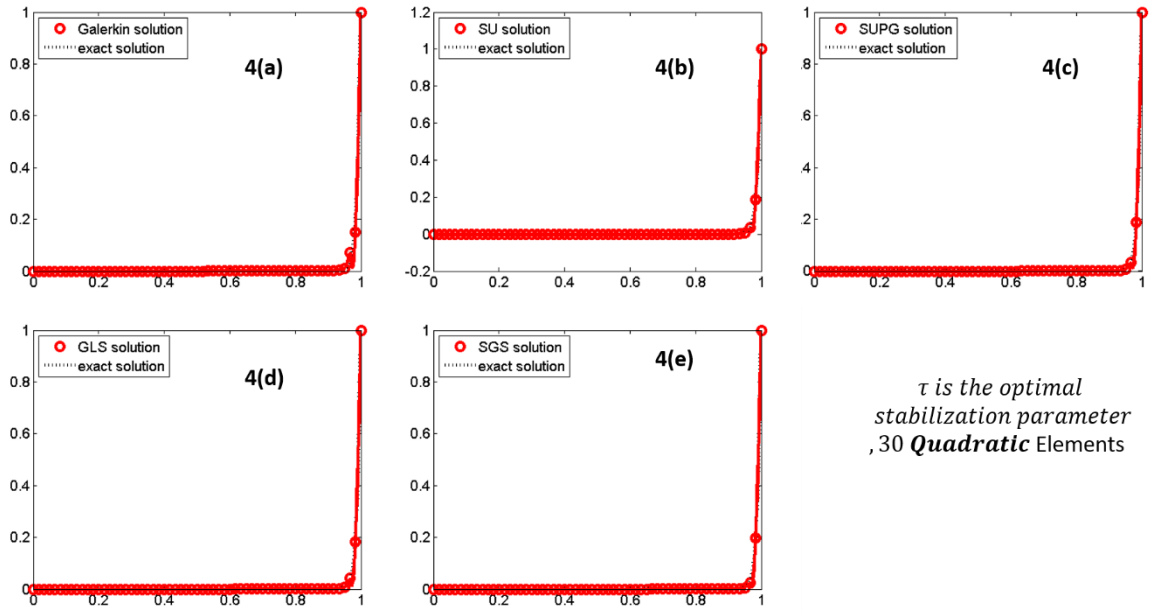

Figure 4: Comparison of solutions from different methods for 30 quadratic elements

GLS Method:-

The stabilised consistent formulation for this method of the given problem is

a (w, u) +c(w,u, a)+$\sum_e \int_0^L (P(w) * \tau * R(u))\, dx = 0$ where R(u)= $au_x - \gamma u_{xx}$

P (w) = $aw_x - \gamma w_{xx}$

The added stabilization term is symmetric. For linear elements it provides similar results as that of SUPG and SU methods. It can be observed in the Figure 1. For quadratic elements it provides much improved results (Figure 4).

Sub-Grid-Scale (SGS) method:

The stabilised consistent formulation for this method of the given problem is

a (w, u) +c(w,u, a)+$\sum_e \int_0^L (P(w) * \tau * R(u))\, dx = 0$ where R(u)= $au_x - \gamma u_{xx}$

P (w) = $aw_x + \gamma w_{xx}$

For linear elements it provides similar results to SUPG, SU and GLS methods as the problem is a convection diffusion problem and $u_{xx} = 0$. It can be observed in the Figure 1. For quadratic elements it provides much improved results (Figure 4).

CODE:-

SUPG (QUADRATIC ELEMENT, P=2)

```
function [K,f] = system_SUPG_p2(tau,tau_c,a,nu,xnode)
% [K,f] = system_SUPG_p2(a,nu,xnode)
% Input:
%   tau, tau_c: stabilization parameters
%   a, nu:      equation parameters
%   xnode :     nodes coordinates
% Gauss points on the reference element [-1,1]
xipg = [-sqrt(15)/5 0 sqrt(15)/5]';
wpg = [5/9 8/9 5/9]';

% Shape functions and its derivatives on the reference element
N_mef   =  [(xipg-1).*xipg/2  1-xipg.^2  (xipg+1).*xipg/2];
Nxi_mef =  [ xipg-1/2 -2*xipg xipg+1/2];
Nxxi_mef=  [ 1 -2 1; 1 -2 1; 1 -2 1];

% Number of nodes and elements
numnp = size(xnode,2);
numel = (numnp-1)/2;

% Number of Gauss points
ngaus = size(wpg,1);

% Allocation of storage for the matrix and vector
K = zeros(numnp,numnp);
f = zeros(numnp,1);

% Matrix of stabilization parameters
% tau_c for the end nodes and tau for the middle one
Tau = diag([tau_c,tau,tau_c]);

% MATRIX AND VECTOR'S CALCULATION
% Loop on the elements
for i = 1:numel
    h = xnode(2*i+1)-xnode(2*i);
    weigth = wpg*h;
    isp = [2*i-1 2*i 2*i+1]; % Global number of the nodes
    % Loop on the Gauss points
    for ig=1:ngaus
        N = N_mef(ig,:);
        Nx = Nxi_mef(ig,:)/h;
        Nxx= Nxxi_mef(ig,:)/(h^2);
        w_ig = weigth(ig);
        x = xnode(2*i) + h*xipg(ig); % x-coordinate of the Gaussian point
        % Assembly
        K(isp,isp) = K(isp,isp) + w_ig*(N'*a*Nx+Nx'*nu*Nx) ...
                                + w_ig*Tau*(a*Nx)'*(a*Nx-nu*Nxx);
        f(isp) = f(isp) + w_ig*(N'+Tau*(a*Nx)')*SourceTerm(x);
    end
end
```

GLS (P=2)

```
MATRIX AND VECTOR CALCULATION
% Loop on the elements
for i=1:numel
    h = xnode(2*i+1)-xnode(2*i);
    weigth = wpg*h;
    isp = [2*i-1 2*i 2*i+1]; % Global number of the nodes in the current element
    % Loop on Gauss points
    for ig=1:ngaus
        N = N_mef(ig,:);
        Nx = Nxi_mef(ig,:)/h;
        Nxx= Nxxi_mef(ig,:)/(h^2);
        w_ig = weigth(ig);
        x = xnode(2*i) + h*xipg(ig); % x-coordinate of the Gauss point
        % Assembly
        K(isp,isp) = K(isp,isp) + w_ig*(N'*a*Nx+Nx'*nu*Nx) ...
                                + w_ig*Tau*(a*Nx-nu*Nxx)'*(a*Nx-nu*Nxx);

        f(isp) = f(isp) + w_ig*(N'+Tau*(a*Nx-nu*Nxx)')*SourceTerm(x);
    end
end
```

SGS (P=2)

```
% MATRIX AND VECTOR CALCULATIONS
% Loop on the elements
for i=1:numel
    h = xnode(2*i+1)-xnode(2*i);
    weigth = wpg*h;
    isp = [2*i-1 2*i 2*i+1]; % Global number of the element's nodes
    % Loop on the Gauss points of the element
    for ig=1:ngaus
        N = N_mef(ig,:);
        Nx = Nxi_mef(ig,:)/h;
        Nxx= Nxxi_mef(ig,:)/(h^2);
        w_ig = weigth(ig);
        x = xnode(2*i) + h*xipg(ig); % x-coordinate of the gaussian point
        % Assembly
        K(isp,isp) = K(isp,isp) + w_ig*(N'*a*Nx+Nx'*nu*Nx) ...
                                + w_ig*Tau*(a*Nx+nu*Nxx)'*(a*Nx-nu*Nxx);
        f(isp) = f(isp) + w_ig*(N'+Tau*(a*Nx+nu*Nxx)')*SourceTerm(x);
    end
end
```

SU (P=2)

```
% MATRIX ANVECTOR CALCULATION
% Loop on the elements
for i=1:numel
    h = xnode(2*i+1)-xnode(2*i);
    weigth = wpg*h;
    isp = [2*i-1 2*i 2*i+1]; % Global number of the element's nodes
    % Loop on the Gauss points
    for ig=1:ngaus
        N = N_mef(ig,:);
        Nx = Nxi_mef(ig,:)/h;
        w_ig = weigth(ig);
        x = xnode(2*i) + h*xipg(ig); % x-coordinate of the Gaussina point
        % Assembly
        K(isp,isp) = K(isp,isp) + w_ig*(N'*a*Nx+Nx'*nu*Nx) ...
                                + w_ig*Tau*(a*Nx)'*(a*Nx);
        f(isp) = f(isp) + w_ig*(N')*SourceTerm(x);
   end
 end
```
