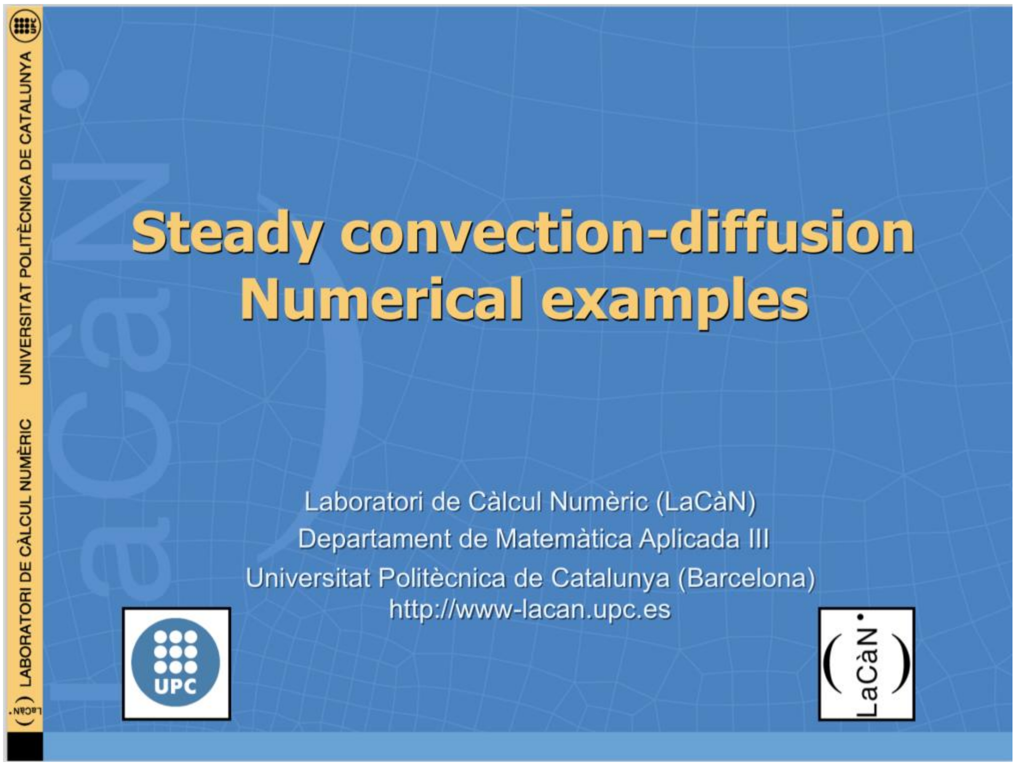

2D Examples

NUMERICAL EXAMPLES

Iberico Leonardo, Juan Diego | Finite Elements in Fluids | 02/04/2018

Example: 2D convection-diffusion equation

The following report it is focus on steady convection-diffusion problem. The aim of the document it is to show the different behavior of the available schemes, to solve a convection-diffusion problem. All the schemes are discretized in the space with the Galerkin formulation.

The problem is located on a square domain, with convection skew to the mesh with an angle of 30º. Discontinuous Dirichlet are imposed on the inlet boundary. On the outlet natural homogeneous boundary conditions are imposed.

EXERCISE FOR STEADY CONVECTION – DIFFUDION EQUATION

- Include the GLS method on the code.

```
elseif method == 3
    % GLS
    aux = N_ig*Xe;
    g_ig = ReactionTerm(aux);
    Ke = Ke + (nu*(Nx'*Nx+Ny'*Ny) + N_ig'*(ax*Nx+ay*Ny)+N_ig'*g_ig*N_ig + ...
        tau*((((ax*Nx+ay*Ny)-nu*(Nxx+Nyy)+g_ig*N_ig)'*((ax*Nx+ay*Ny)-nu*(Nxx+Nyy)+g_ig*N_ig)))...
        *dvolu;
```

The implementation of the code was done in the matlab function FEM_system.m as a third method, including already the reaction term.

- Modify the code to solve a steady convection-diffusion reaction problem with zero Dirichlet boundary conditions on the outlet boundary. Compare the methods behavior to the one observed when Neumann boundary conditions were imposed.

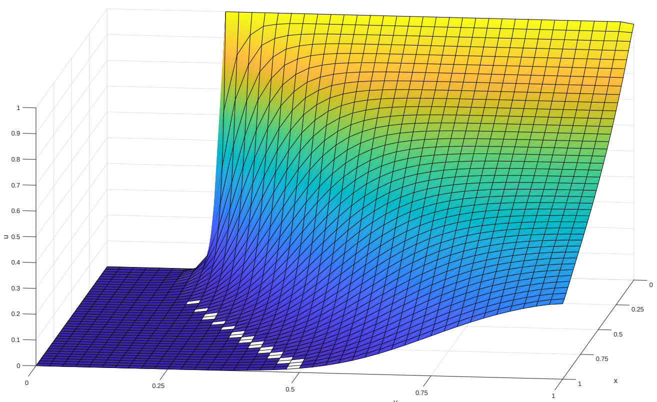

Figure 2 SUPG scheme with Neumann b.c

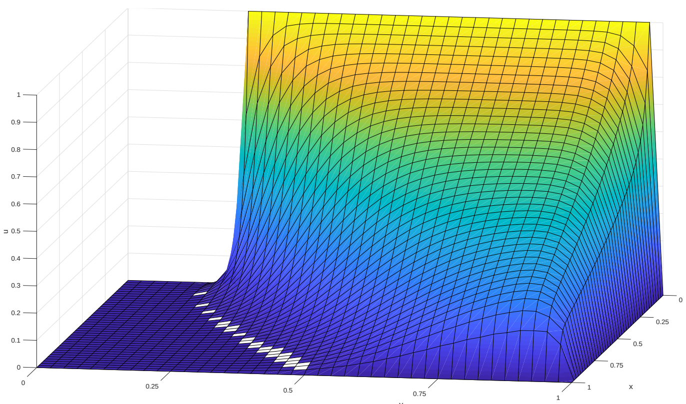

Figure 1 SUPG scheme with zero b.c on the Outflow

After the modifications on the boundary conditions and force the outlet flow to take the value 0, we can appreciate on the creation of a boundary layer on the outflow sides.

- Solve the problem for

  A convection-reaction dominated case with $a = \frac{1}{2},\ \nu = 0.01,\ \sigma = 1$

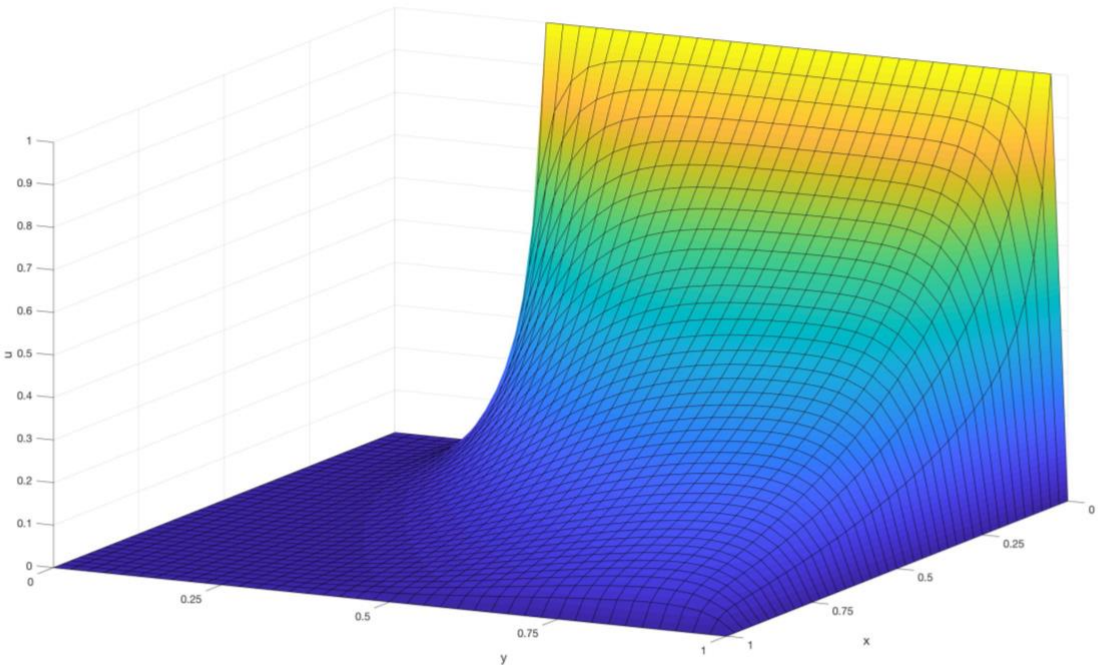

Figure 3 SUPG scheme

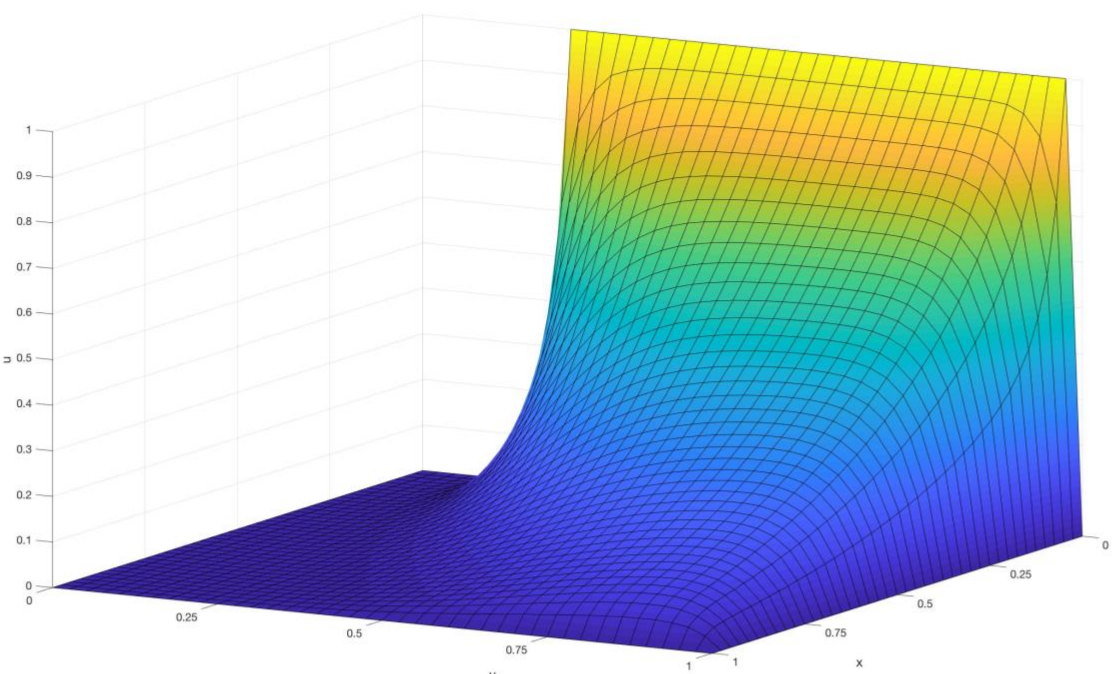

Figure 4 GLS scheme

- A reaction dominated case with $a = 0.001$, $\nu = 0.01$, $\sigma = 1$

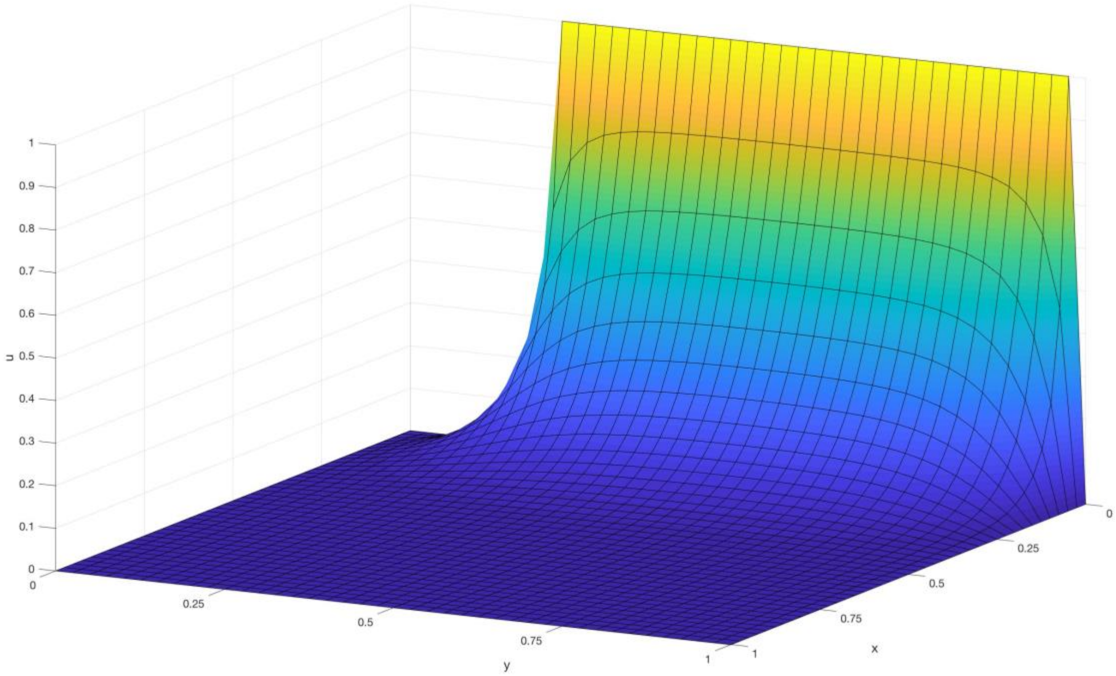

Figure 5 SUPG scheme

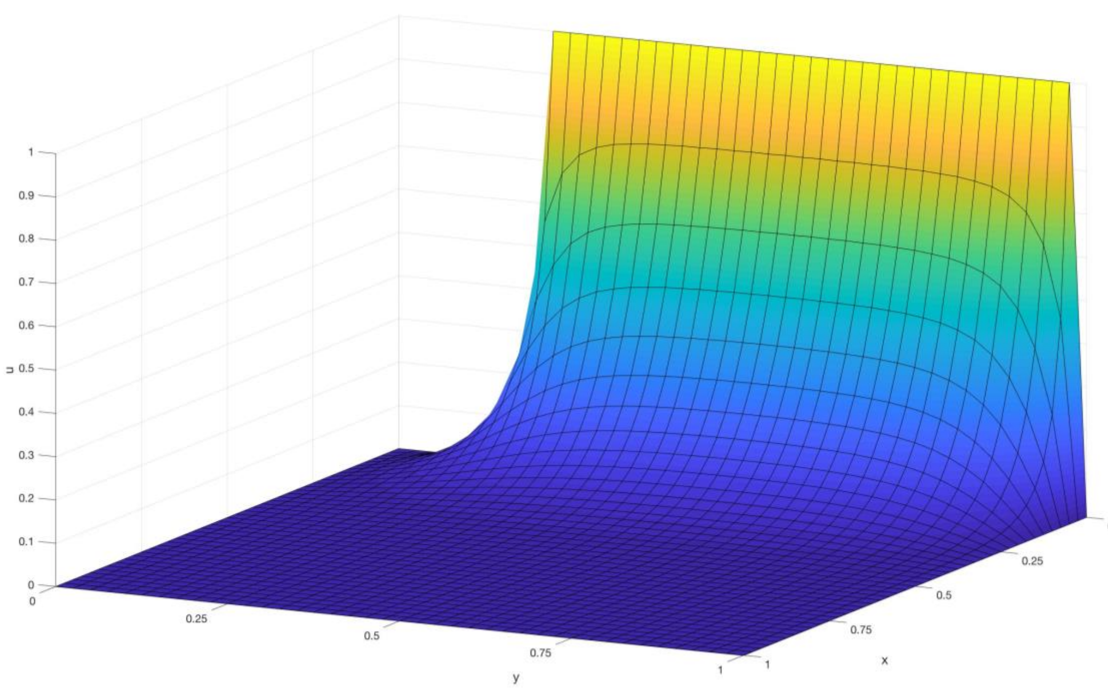

Figure 6 GLS scheme

On one side, for the convection-diffusion problem the plots show the transport of the flow until reach the boundary layer taking velocity as value 0. The transport of the particles is decreasing by the introduction of the diffusion term.

On the other side, for the reaction dominated case, the transport decrease in a very steep form, reaching velocity 0 far from the boundary layer.

Code implementation:

Implementation boundary conditions (for 2nd order elements)

```
% BOUNDARY CONDITIONS
% Boundary conditions are imposed using Lagrange multipliers
if p == 1
    nodes_y0 = [1:nx+1]';                              % Nodes on the boundary y=0
    nodes_x1 = [2*(nx+1):nx+1:(ny+1)*(nx+1)]' ;      % Nodes on the boundary x=1
    nodes_y1 = [ny*(nx+1)+nx:-1:ny*(nx+1)+1]' ;      % Nodes on the boundary y=1
    nodes_x0 = [(ny-1)*(nx+1)+1:-(nx+1):nx+2]';      % Nodes on the boundary x=0
elseif p == 2
    nodes_y0 = [1:2*nx+1]';                              % Nodes on the boundary y=0
    nodes_x1 = [2*(2*nx+1):2*nx+1:(2*ny+1)*(2*nx+1)]' ;      % Nodes on the boundary x=1
    nodes_y1 = [2*ny*(2*nx+1)+2*nx:-1:2*ny*(2*nx+1)+1]' ;      % Nodes on the boundary y=1
    nodes_x0 = [(2*ny-1)*(2*nx+1)+1:-(2*nx+1):2*nx+2]';      % Nodes on the boundary x=0
end

% nodes on which solution is u=1
nodesDir1 = nodes_x0( X(nodes_x0,2) > 0.2 );
% nodes on which solution is u=0
%nodesDir0 = [nodes_x0( X(nodes_x0,2) <= 0.2 ); nodes_y0]; % B.C default and the modified
nodesDir0 = [nodes_x0( X(nodes_x0,2) <= 0.2 ); nodes_y0; nodes_x1; nodes_y1 ];
```

Implementation for triangular elements linear and quadratic:

```
elseif elem == 1
    if p == 1
        N = [1-xi-eta, xi, eta];
        Nxi = [-ones(size(xi)), ones(size(xi)), zeros(size(xi))];
        Neta = [-ones(size(xi)), zeros(size(xi)), ones(size(xi))];
        N2xi = zeros(size(xi));
        N2eta = zeros(size(xi));
    elseif p == 2
        N = [ xi.*(2*xi-1), eta.*(2*eta-1), (1-xi-eta).*(2.*(1-xi-eta)-1),...
            4.*xi.*eta, 4.*eta.*(1-xi-eta), 4.*(1-xi-eta).*xi];
        Nxi = [ 4.*xi - 1, zeros(size(xi)), 4.*eta + 4.*xi - 3, 4.*eta, -4.*eta, 4 - 8.*xi - 4.*eta];
        Neta = [ zeros(size(xi)), 4.*eta - 1, 4.*eta + 4.*xi - 3, 4.*xi, 4 - 4.*xi - 8.*eta, -4.*xi];
        N2xi = [ 4.*ones(size(xi)), 0*ones(size(xi)), 4.*ones(size(xi)), 0.*ones(size(xi)), 0.*ones(size(xi)), -8.*ones(size(xi))];
        N2eta = [  0.*ones(size(xi)), 4.*ones(size(xi)), 4.*ones(size(xi)), 0*ones(size(xi)), -8.*ones(size(xi)), 0.*ones(size(xi))];
    else
        error('not available interpolation degree')
    end
```

EXERCISE FOR UNSTEADY CONVECTION EQUATION

For this example, we consider a 2D homogeneous convection equation with initial condition and homogeneous Dirichlet conditions on the inlet boundary.

- Implement the code with a scheme of high order: The implement method of r that problem is Fourth order 2 step.

```
elseif meth == 8
    % 1st step implementation with delta u = containing u_tilde
    A1 = M;
    B1 = (1/3)*C*dt - (1/12)*K*dt^2 - (1/3)*Mo*dt + (1/12)*Co*dt^2;
    f1 = (1/3)*v1*dt + (1/12)*v2*dt^2 - (1/12)*vo*dt^2;
    % 2nd step implementation with dealta u = containing u_n+1
    A2 = M;
    B2 = C*dt - Mo*dt; %containing u
    f2 = v1*dt + 0.5*v2*dt^2 - 0.5*vo*dt^2; %containing s
    C2 = -0.5*K*dt^2 + 0.5*Co*dt^2; %containing u_tilde
```

The transient solution is implemented as it follows:

```
elseif meth == 8   % 2-step method
    btot = [B1*u(:,n)+ f1; bccd];
    aux =  U1\(L1\btot);
    u_m =  u(:,n) + aux(1:numnp);
    btot = [B2*u(:,n) + C2*u_m + f2; bccd];
    aux  = U2\(L2\btot);
    u(:,n+1) = u(:,n) + aux(1:numnp);
```

- Discuss the behavior of the methods.

If we compare the behavior of the different methods, we can notice the significant numerical errors that appear in the schemes of 2 second order accuracy.

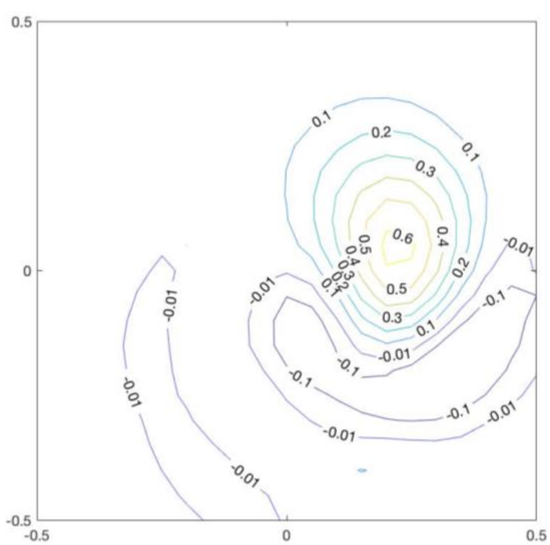

Figure 7 LW-FD scheme, problem 1

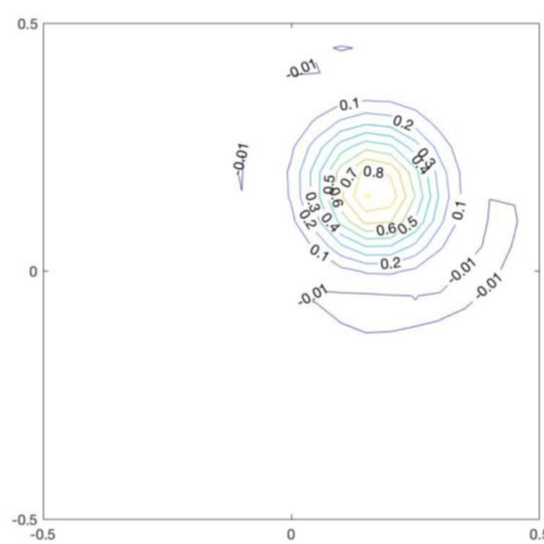

Figure 8 TG4