INTERNATIONAL CENTRE FOR
NUMERICAL METHODS IN ENGINEERING

UNIVERSITAT POLITÈCNICA DE CATALUNYA

MASTER OF SCIENCE IN COMPUTATIONAL MECHANICS

Finite Element in Fluids

Homework 2

Eugenio José Muttio Zavala

February 20, 2019

Submitted To:
Prof. Antonio Huerta
Prof. Pablo Saez

1 Steady Convection Diffusion Equation in 1D

1.1 Strong Form

Considering the problem in which is implied the transport of fluid particles represented by a scalar quantity $u = u(x)$ in a domain Ω with smooth boundary Γ. The boundary is composed by a zone Γ_D with prescribed transport value u, named "Dirichlet Boundary ", and another zone Γ_N with prescribed diffusive flux known as "Neumann Boundary". The boundary value problem is defined by the equations:

$$\mathbf{a} \cdot \nabla u - \nabla \cdot (\nu \nabla u) = s \qquad in \qquad \Omega \tag{1.1}$$

$$u = u_D \qquad in \qquad \Gamma_D \tag{1.2}$$

$$\nu \frac{\partial u}{\partial n} = u_N \qquad in \qquad \Gamma_N \tag{1.3}$$

Where:

- u - is the scalar unknown.
- a - is the convection velocity.
- ν - is the diffusion coefficient.
- s - is a source term.

Notice that $\nu > 0$ the problem becomes as the solution of an elliptic partial differential equation. In contrast if the diffusion coefficient is equal to zero then the problem is *pure convective* and the PDE is hyperbolic.

1.2 Weak Form

In order to solve the problem numerically by using the Finite Element Method, it is necessary to obtain the integral or "weak" form of the convection-diffusion equation. By choosing the Galerkin approximation approach, the first step is operate the expression by multiplicating a test function w and integrate in the full domain:

$$\int_\Omega w(\mathbf{a} \cdot \nabla u) d\Omega - \int_\Omega w \nabla \cdot (\nu \nabla u) d\Omega = \int_\Omega w s d\Omega$$

Because of the second derivatives involved in the above equation, there is a lack of simmetry which leads to an non-symmetric expression of the algebraic equation system from the discretization process. In that sense it is convenient to add a restriction on w and get rid of the second derivative on u. For that reason is necessary to do a integation by parts, which formula is:

$$\int_\Omega g(\nabla \mathbf{f})d\Omega = -\int_\Omega \nabla g \cdot \mathbf{f} d\Omega + \int_{\Gamma_N} g\mathbf{f}\cdot\mathbf{n}d\Gamma \tag{1.4}$$

Then, the next step is to use this formula in the steady transport equation:

$$\int_\Omega w(\mathbf{a}\cdot\nabla u)d\Omega + \int_\Omega \nabla w\cdot(\nu\nabla u)d\Omega = \int_\Omega wsd\Omega + \int_{\Gamma_N} wu_N d\Gamma \tag{1.5}$$

It is important to remark that the integration by parts of the diffusion term allows us to naturally introduce the prescribed flux condition on the Neumann boundary. Now, the equation 1.5 can be presented in a compact form:

$$\mathbf{a}(w,u) + \mathbf{c}(w,u,\mathbf{a}) = (w,s) + (w,u_N)_{\Gamma_N} \tag{1.6}$$

where:

$$\mathbf{a}(w,u) = \int_\Omega \nabla w\cdot(\nu\nabla u)d\Omega$$

$$\mathbf{c}(w,u,\mathbf{a}) = \int_\Omega w(\mathbf{a}\cdot\nabla u)d\Omega$$

$$(w,s) = \int_\Omega wsd\Omega$$

$$(w,u_N)_{\Gamma_N} = \int_{\Gamma_N} wu_N d\Gamma$$

1.3 DISCRETIZATION

To perform the numerical solution of the equations explained before, it is important to remark that the type of approximation is implemented by Galerkin method. Having in mind this, it can be consider that the main ingredients to perform the spatial discretization of the convection-diffusion problem is already given. Then, Galerkin formulation works by restricting the weak form to the approximate solution u^h written as:

$$u(x)^h = \sum N_A(x)u_A + N_D(x)u_D \tag{1.7}$$

where N_A is the shape function associated with the node number A and u_A is the nodal unknown. Morever, as Galerkin shows, the test functions w^h are defined by the same shape functions. This strategy allows to convert the weak form in a set of equations that govern the nodal values of the discrete solution of the convection-diffusion problem:

$$(\mathbf{C}+\mathbf{K})\mathbf{u} = \mathbf{f} \tag{1.8}$$

The equation 1.8 contains **u** which is the vector of nodal values, **C** the convection matrix and **K** the diffusion matrix. Both matrices are obtained by topological assembly of each finite element considered in the discretization, then the convection matrix is:

$$\mathbf{C}^{\mathbf{e}}_{\mathbf{ab}} = \int_{\Omega^e} N_a(\mathbf{a}\cdot\nabla N_b)d\Omega \tag{1.9}$$

Diffusion matrix is defined as:

$$\mathbf{K}^{\mathbf{e}}_{\mathbf{ab}} = \int_{\Omega^e} \nabla N_a\cdot(\nu\nabla N_b)d\Omega \tag{1.10}$$

The r.h.s vector considers the contribution of the source term s, and it may includes the information given previously by the Neumann boundary, which are the prescribed fluxes:

$$\mathbf{f}^{\mathbf{e}}_{\mathbf{a}} = \int_{\Omega^e} N_a s d\Omega + \int_{\Omega^e} N_a u_N d\Omega \tag{1.11}$$

1.3.1 Galerkin Linear Approximation

The weak form is now discretized in a uniform mesh of linear elements of size "h". As is one dimension, the global numbering of the nodes is consecutive, and the local numbering is denoted as 1 and 2. The shape functions of a linear element are:

$$N_1(\xi) = \frac{1}{2}(1-\xi) \tag{1.12}$$

$$N_2(\xi) = \frac{1}{2}(1+\xi) \tag{1.13}$$

where ξ is the normalized coordinate, $-1 \leq \xi \leq 1$. At any interior of the element has a solution interpolation and the geometry interpolation given by the next equations respectively:

$$u(\xi) = N_1(\xi)u_1 + N_2(\xi)u_2 \tag{1.14}$$

$$u(\xi) = N_1(\xi)x_1 + N_2(\xi)x_2 \tag{1.15}$$

and thus,

$$\frac{\partial N_b}{\partial x} = \frac{\partial N_b}{\partial \xi}\frac{\partial \xi}{\partial x} = \frac{2}{h}\frac{\partial N_b}{\partial \xi} \qquad for \quad b = 1,2. \tag{1.16}$$

The evaluation of the matricial system of equations require a numerical integration, that is the relevance of transform to a normalized coordinate system. Using Gauss quadratures at the normalized points, it can transform the integral form as a sum of products defined by the reference element points $z_g = (\xi_g, \eta_g)$ and weights ω_g, for example the source term is integrated as:

$$\int_{\Omega^e} N_a s d\Omega \approx \sum_{g=1}^{n_{gauss}} N_i(z_g)s(x(z_g))\|J(z_g\|\cdot\omega_g$$

The Galerkin approach to solve numerically the 1D steady transport equation by FEM is implemented in Matlab with the methodologies explained above, so adding these ingredients into the code some exercises can be analyzed.

PROBLEM STATEMENT

Solve the 1D convection-diffusion equation with constant coefficients. The source term is equal to $s = 0$, and the Dirichlet boundary conditions are $u_0 = 0$ and $u_1 = 1$. Different values of the coefficients of convection and diffusion will be tested and the discretization is constant with 10 linear elements.

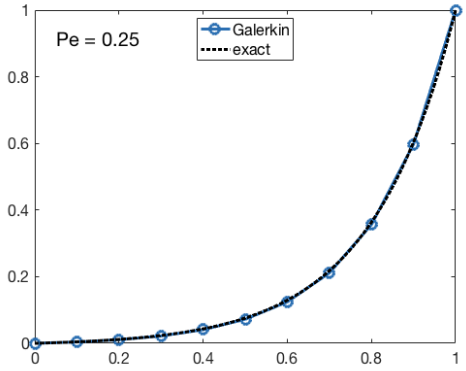

Figure 1.1: $a = 1$, $\nu = 0.2$, 10 linear elements.

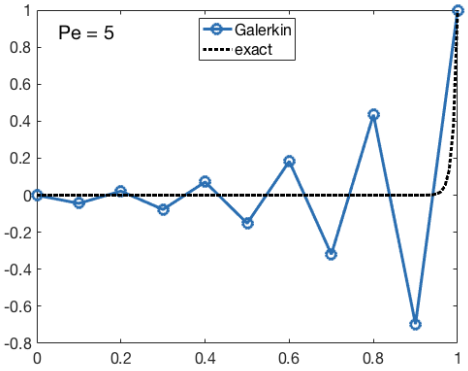

Figure 1.3: $a = 1$, $\nu = 0.01$, 10 linear elements.

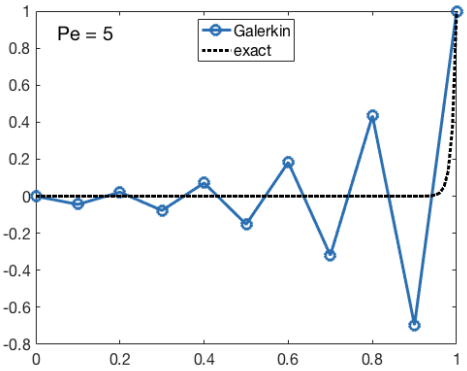

Figure 1.2: $a = 20$, $\nu = 0.2$, 10 linear elements.

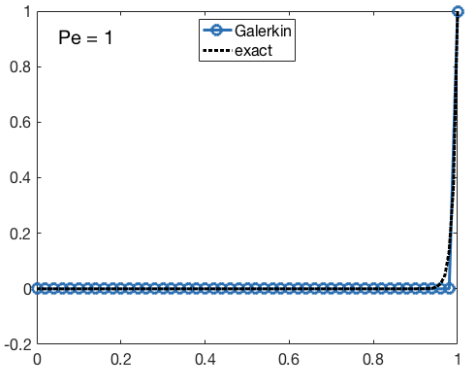

Figure 1.4: $a = 1$, $\nu = 0.01$, 50 linear elements.

Discussion 1

The above experiments can show some interesting facts about the Galerkin method when is coded to solve numerically the steady transport equation. The figure 1.1 shows that in a problem with balanced diffusion compared with the convection, the solution is useful even with only ten elements. The problem starts when the convection is considerable greater than diffusion like the figure 1.2 and 1.3 where the solution at nodes starts to oscillate. Increasing the number of elements the solution get better as is expected when using FEM methodology.

The importance between the convective and diffusive effects can be measured by the Péclet number:

$$P_e = \frac{\|a\| h}{2\nu} \tag{1.17}$$

This number depends on the convective and diffusive coefficients, but also in the mesh used in the experiment. As is presented in the graphs, the Péclet number is directly involved in the behavior of the numerical solution. The first graph has a Péclet $P_e = 0.25$ and performs good, meanwhile the figure 1.2 and 1.3 have a Péclet $P_e = 5$ even though the coefficients are different, but because of the same ratio the result is oscillatory. With this results, one can notes that Galerkin solution is corrupted by non-physical oscillations that no corresponds with the initial problem when Péclet is larger than one. The next try will be solve the equation by using FEM but with a quadratic elements approximation.

1.3.2 Galerkin Quadratic Approximation

The methodology implementing quadratic elements is more or less the same by using linear elements, the main difference it is the discretization with quadratic shape functions, the equation matrices will be bigger and the numerical integration will vary in coefficients in order to use the same approximation. Considering an element with end nodes 1 and 3, and a mid-side node 2, and the normalized coordinate $-1 \leq \xi \leq 1$, the shape functions of the elements are:

$$N_1(\xi) = \frac{1}{2}\xi(\xi - 1) \tag{1.18}$$

$$N_2(\xi) = (1 - \xi^2) \tag{1.19}$$

$$N_3(\xi) = \frac{1}{2}\xi(\xi + 1) \tag{1.20}$$

At any interior of the element has a solution interpolation and the geometry interpolation given by the next equations respectively:

$$u(\xi) = N_1(\xi)u_1 + N_2(\xi)u_2 + N_3(\xi)u_3 \tag{1.21}$$
$$x(\xi) = N_1(\xi)x_1 + N_2(\xi)x_2 + N_3(\xi)x_3 \tag{1.22}$$

If a uniform mesh is used, the middle node is located at $x_2 = \frac{1}{2}(x_1 + x_3)$. Then, if the characteristic size h (where h is considered the distance between nodes *not the element size*), the following relations hold between the normalized and physical coordinates: $dx = hd\xi$, then:

$$\frac{\partial N_b}{\partial x} = \frac{\partial N_b}{\partial \xi}\frac{\partial \xi}{\partial x} = \frac{1}{h}\frac{\partial N_b}{\partial \xi} \qquad for \quad b = 1,2,3. \tag{1.23}$$

Code Implementation 1

The Matlab code used in the Galerkin linear approximation is the same for the quadratic approximation, just is necessary to add a modification in the nodes assignment at the time of the discretization. In other words, instead of having a connectivity matrix given by T=[1:nPt-1; 2:nPt] where nPt is the number of nodes of the mesh, the quadratic connectivity matrix has the form T=[2*I-1; 2*I; 2*I+1] for each element in the discretization. The numerical integration is modified too, changing the Gauss quadrature values to consider a integration of order 3, and finally adding a conditional to select if the approximation is linear o quadratic. The same experiments done before were tested with this higher order element.

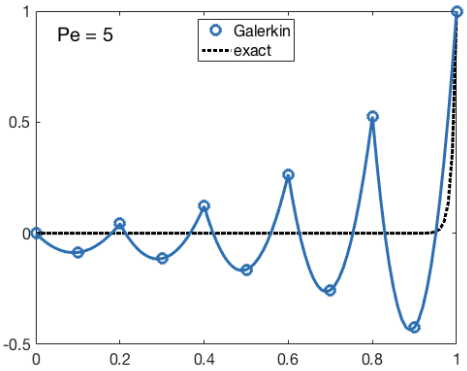

Figure 1.5: $a = 20$, $\nu = 0.2$, 5 quadratic elements.

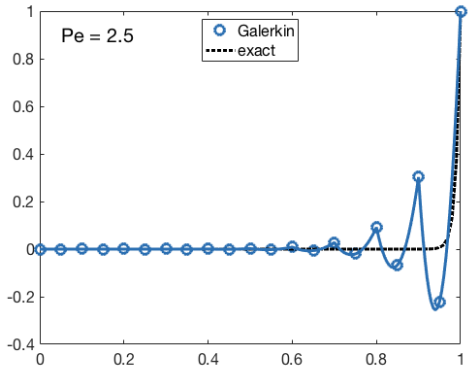

Figure 1.6: $a = 1$, $\nu = 0.01$, 10 quadratic elements.

Discussion 2

Using higher order elements as the quadratic approximation, it could be expected a better solution compared with the linear. This convergence behavior only happens in problems where the convection-diffusion effects are balanced. But in fact, comparing with the same experiments as before, the quadratic element also has problems in convection dominated problems, mainly because Galerkin delivers two types of nodal equations representing the discrete counterpart of the convection-diffusion equation. As seen in the figures, the Péclet number above 1 indicates node oscillation in the solution. By decreasing the size of the discretization could performs better but in order to obtain a useful result it will be needed a very fine mesh that can be computational expensive.

1.3.3 Source Term

The next test that is recommended to attend using Galerkin approach with linear and quadratic elements is to add a source term which is not constant as the previous experiments. The function of source term is:

$$s = 10e^{-5x} - 4e^{-x} \tag{1.24}$$

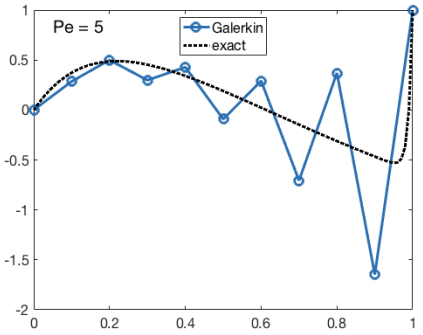

Figure 1.7: $a = 1$, $\nu = 0.01$, 10 linear elements, source term $s = 10e^{-5x} - 4e^{-x}$.

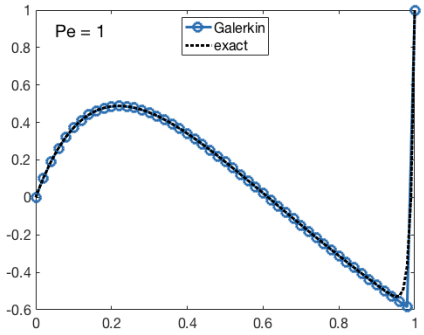

Figure 1.8: $a = 1$, $\nu = 0.01$, 50 linear elements, source term $s = 10e^{-5x} - 4e^{-x}$.

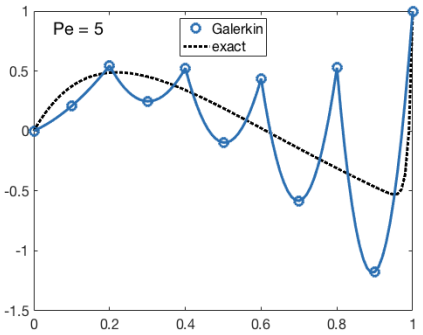

Figure 1.9: $a = 1$, $\nu = 0.01$, 5 quadratic elements $s = 10e^{-5x} - 4e^{-x}$.

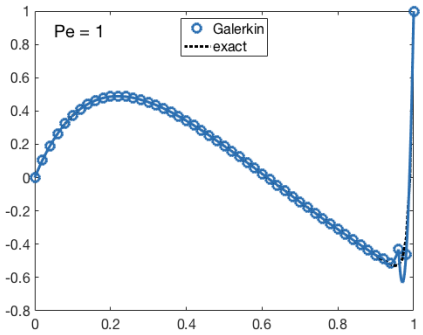

Figure 1.10: $a = 1$, $\nu = 0.01$, 25 quadratic elements $s = 10e^{-5x} - 4e^{-x}$.

Discussion 3

In that case, it can be shown that Galerkin shows defficiencies again, first because of the Peclét number is over 1, which causes oscillations in the solution nodes. Second, the soure term now is not a constant value, and as it was explained in the theory, using this type of function it could be added unwanted truncation errors due the spatial discretization of the source term. So, as can it can be observed in the graphs, it is needed 50 and 25 linear and quadratic elements respectively to reach the Péclet number equal to 1 and obtain a better solution.

1.4 Streamline Upwind SU

As seen in the previous experiments, Galerkin presents deficiencies in convection dominated problems, so the general idea to avoid this problem is adding "more weight" to the terms associated with the transport in the upwind direction. Solving the convection diffusion problem by employing linear elements it is possible to formulate an optimal upwind technique producing exact values in the nodes, modifying the initial equation as:

$$au_x - (\nu + \bar{\nu})u_{xx} = 0 \qquad with \qquad \bar{\nu} = \beta\frac{ah}{2} \tag{1.25}$$

where the magnitude of the added diffusion, $\bar{\nu}$, is governed by the free parameter $\beta(0 \leq \beta \leq 1)$, which value can be calculated optimally by:

$$\beta = \coth(P_e) - \frac{1}{P_e} \tag{1.26}$$

If the discretization proposed is by using quadratic elements, the formulation is more complicated due the difference between the approximation solutions given by the corner nodes and the mid-side node, seen in the previous section using Galerkin. At the mid-side nodes, the optimal value of the added diffusivity is given by the same formula of the linear elements of the equation . The artificial viscosity that needs to be added at the corner nodes is:

$$\beta_{corner} = \frac{(\coth(P_e) - 1/P_e) - (\cosh(P_e))^2(\coth(2P_e) - 1/(2P_e))}{1 - (\cosh(P_e))^2/2} \tag{1.27}$$

Code Implementation 2

Now, with this different approach it is possible to implement a solution for the transport equation which is more stable than the original Galerkin. So, in the Matlab code it is needed to add a conditional which selects the type of approximation (linear or quadratic) to choose the β that is needed to perform the matrix assembling. So if the quadratic element is requested, the code activates the formula given before to add the corner nodes.

Now consider some of the experiments done by Galerkin but using the streamline upwind technique, using a constant source term equal to zero and another one with the form $s = 10e^{-5x} - 4e^{-x}$:

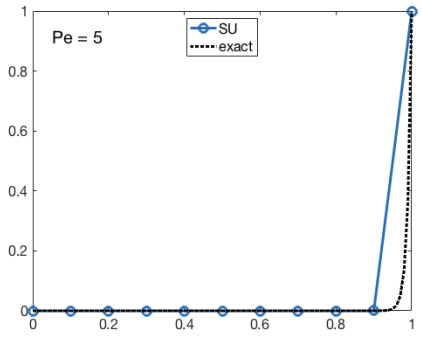

Figure 1.11: $a = 1$, $\nu = 0.01$, 10 linear elements, source term $s = 0$.

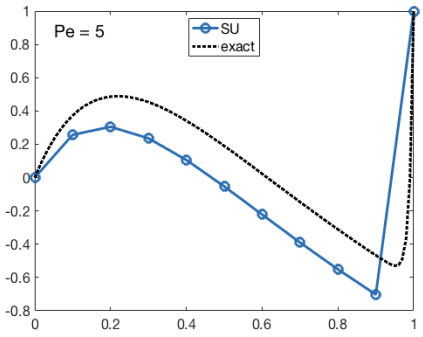

Figure 1.13: $a = 1$, $\nu = 0.01$, 10 linear elements, source term $s = 10e^{-5x} - 4e^{-x}$.

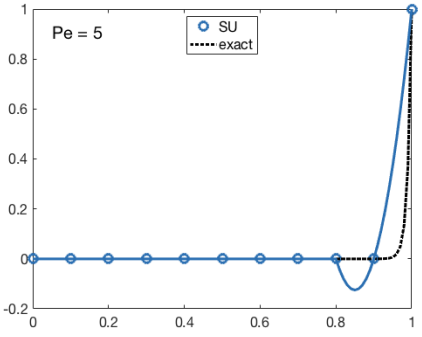

Figure 1.12: $a = 1$, $\nu = 0.01$, 5 quadratic elements $s = 0$.

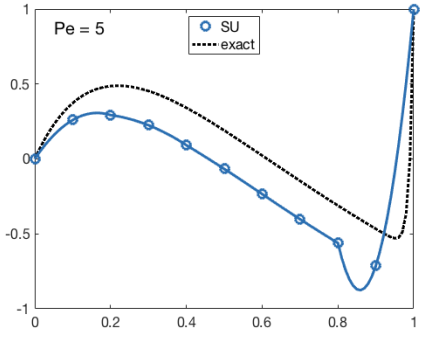

Figure 1.14: $a = 1$, $\nu = 0.01$, 5 quadratic elements $s = 10e^{-5x} - 4e^{-x}$.

Discussion 4

Compared by the Galerkin experiments, the solution is completely different and much better, it can be seen the node values are practically the same as the exact solution, even when using the linear approximation. As can be seen, the results are not acceptable by using a source term that is not a constant number. This behavior is due to the artificial viscosity added to the discrete problem, in which the SU technique is based.

1.5 Streamline Upwind Petrov Galerkin SUPG

In order to stabilize the convective term in a consistent manner, first consider the steady convection-diffusion-reaction equation:

$$\mathbf{a}\cdot\nabla u - \nabla\cdot(\nu\nabla u) + \sigma u = s \qquad in \qquad \Omega \tag{1.28}$$

Now, consider the residual $\mathcal{R}(u)$ of the differential equation as:

$$\mathcal{R}(u) = \mathbf{a}\cdot\nabla u - \nabla\cdot(\nu\nabla u) + \sigma u - s = \mathcal{L}(u) - s \qquad in \qquad \Omega \tag{1.29}$$

Then, the general form of the consistent stabilization techniques is:

$$\begin{aligned} \mathbf{a}(w,u) + \mathbf{c}(w,u,\mathbf{a}) + (w,\sigma u) + \\ + \sum_e \int_{\Omega^e} \mathcal{P}(w)\tau\mathcal{R}(u)d\Omega = (w,s) + (w,u_N)_{\Gamma_N} \end{aligned} \tag{1.30}$$

where $\mathcal{P}(w)$ is a certain operator depending on the technique and τ is the stabilization parameter or intrinsic time.

The Streamline Upwind Petrov Galerkin technique is defined by:

$$\mathcal{P}(w) = \mathbf{a}\cdot\nabla u \tag{1.31}$$

Then, the discrete problem that must be solved is:

$$\begin{aligned} \mathbf{a}(w,u) + \mathbf{c}(w,u,\mathbf{a}) + (w,\sigma u) + \\ + \sum_e \int_{\Omega^e} (\mathbf{a}\cdot\nabla w^h)\tau[\mathbf{a}\cdot\nabla u - \nabla\cdot(\nu\nabla u) + \sigma u - s]d\Omega \\ = (w,s) + (w,u_N)_{\Gamma_N} \end{aligned} \tag{1.32}$$

where the stabilization parameter τ is:

$$\tau = \frac{\bar{\nu}}{\|a\|^2} \tag{1.33}$$

and $\bar{\nu} = \beta ah/2$ for 1D case. For the linear approximation is noticeable that the stabilization term in the conevction-diffusion equation reduces to:

$$\sum_e \int_{\Omega^e} (\mathbf{a}\cdot\nabla w^h)\tau[\mathbf{a}\cdot\nabla u - s]d\Omega \tag{1.34}$$

Code Implementation 3

Considering this in the Matlab implementation, we can observe that the only modification from the **linear approximation** from the SU code will be add in the "force vector" the corresponding source term multiplied by $(\mathbf{a}\cdot\nabla w^h)\tau$. The difference operator disappears because of the linearity of the approximation.

However, in the **quadratic approximation** it is necessary to consider the stabilization term as:

$$\sum_e \int_{\Omega^e} (\mathbf{a}\cdot\nabla w^h)\tau[\mathbf{a}\cdot\nabla u - \nabla\cdot(\nu\nabla u) - s]d\Omega \tag{1.35}$$

Similar to the previous techniques, for the **quadratic approximation** the mid-side node and the corner nodes will be different respect to the β parameter, in which value for the mid-side nodes is the same as the equation 1.27 and for the corner nodes is:

$$\beta_{corner} = \frac{(2P_e-1)+(-6P_e+7)e^{-2P_e}+(-6P_e-7)e^{-4P_e}+(2P_e+1)e^{-6P_e}}{(P_e+3)+(-7P_e-3)e^{-2P_e}+(7P_e-3)e^{-4P_e}-(P_e+3)e^{-6P_e}} \tag{1.36}$$

Now, in order to observe the behavior of SUPG it is good to compare the same exercises solved by the SU technique in which the source term is constant and when is not.

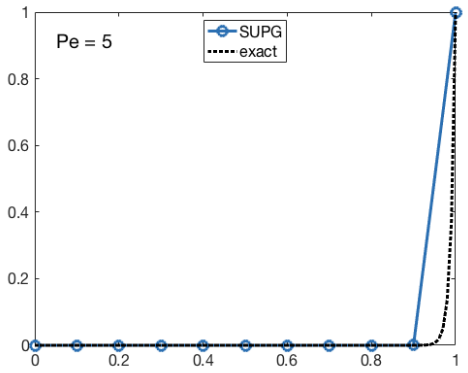

Figure 1.15: $a = 1$, $\nu = 0.01$, 10 linear elements, source term $s = 0$.

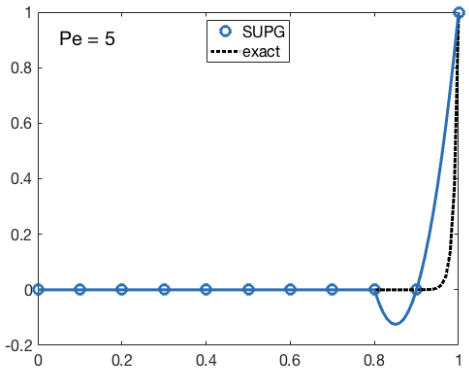

Figure 1.16: $a = 1$, $\nu = 0.01$, 5 quadratic elements $s = 0$.

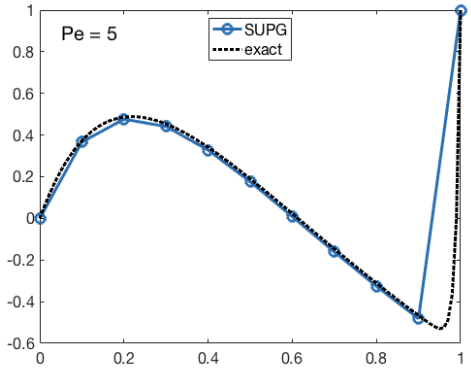

Figure 1.17: $a = 1$, $\nu = 0.01$, 10 linear elements, source term $s = 10e^{-5x} - 4e^{-x}$.

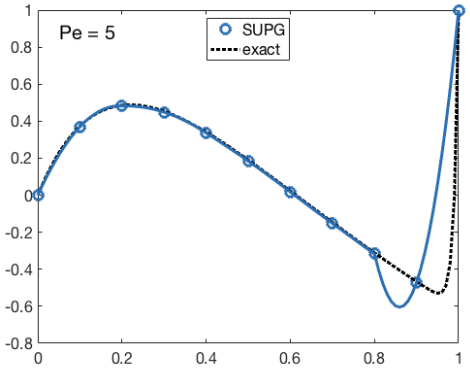

Figure 1.18: $a = 1$, $\nu = 0.01$, 5 quadratic elements $s = 10e^{-5x} - 4e^{-x}$.

Discussion 5

As it can be seen in the graphs, SUPG performs as good as SU when the source term is constant, but the great advantage from the other two methods is reached at looking the solution with a source term with a function. With less quantity of elements, the solution at the nodes is practically the same as the exact. The interpolation of the last part is not very good but adding a few more elements it can be reduced this error.

1.6 Galerkin/Least-Squares Method GLS

The GLS technique is defined by imposing the stabilization term is an element-by-element weighted least squares formulation of the differential equation. This corresponds to the following expression suggested applied to the test function:

$$\mathcal{P}(w) = \mathcal{L}(w) = \mathbf{a}\cdot\nabla w - \nabla\cdot(\nu\nabla w) + \sigma w \tag{1.37}$$

With this definition, the weak form that must be solved is: find u^h such that:

$$\begin{aligned}\mathbf{a}(w,u) + \mathbf{c}(w,u,\mathbf{a}) + (w,\sigma u) + \sum_e \int_{\Omega^e} \mathcal{L}(w)\tau[\mathcal{L}(u) - s]d\Omega \\ = (w,s) + (w,u_N)_{\Gamma_N}\end{aligned} \tag{1.38}$$

in which the stabilization term that affects the l.h.s is symmetric, this is the advantage of this stabilization technique.

Code Implementation 4

The major changes in the implementation of the code compared to SUPG, is the second term of $\mathcal{P}(w)$ which affects when quadratic elements are employed, if not the GLS is exactly the same as the linear SUPG and for no reaction problems. The next graphs show the same experiments done by the previous methods:

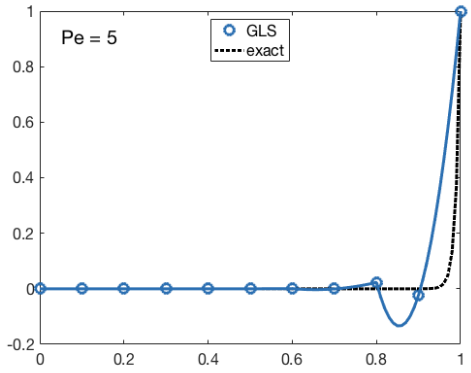

Figure 1.19: $a = 1$, $\nu = 0.01$, 5 quadratic elements, source term $s = 0$.

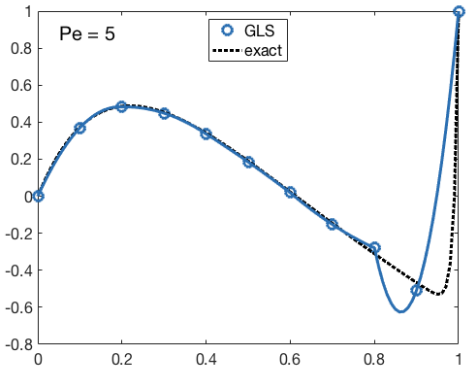

Figure 1.20: $a = 1$, $\nu = 0.01$, 5 quadratic elements $s = 10e^{-5x} - 4e^{-x}$.

1.7 Conclusion

Final Discussion 1

Finite Element Method is a powerful methodology, which formulated with the most general and famous scheme to approximate solutions, named "Galerkin", can obtain *almost* exact solutions for problems that involved basic partial differential equations and specific conditions. But, in some cases as the *"Steady Convection-Diffusion Equation in Fluids"*, Galerkin alone can not solve adequately the problem. That is because of the physics and mathematical nature of the equation that relates another type of PDE when the diffusion parameter is tending to zero. For that reason, the numerical techniques of stabilization as SU, SUPG and GLS add interesting operators that helps Galerkin to counteract this behavior obtaining a variety of useful numerical tools and therefore better results.