FINITE ELEMENT SUPG PARAMETERS COMPUTED FROM LOCAL EDGE MATRICES FOR COMPRESSIBLE FLOWS

Lucia Catabriga
Department of Computer Science (DI)
Federal University of Espírito Santo (UFES)
Av. Fernando Ferrari, s/n, Goiabeiras, 29060-900, Vitória, ES, Brazil
luciac@inf.ufes.br

Alvaro L. G. A. Coutinho
Department of Civil Engineering - COPPE
Federal University of Rio de Janeiro (UFRJ)
Caixa Postal 68506, 21945 970, Rio de Janeiro, RJ, Brazil
alvaro@nacad.ufrj.br

Tayfun E. Tezduyar
Mechanical Engineering, Rice University – MS 321
6100 Main Street, Houston, Texas 77005, USA
tezduyar@rice.edu

Abstract. *For the SUPG formulation of inviscid compressible flows, we present stabilization parameters defined based on the edge-level matrices and the degree-of-freedom submatrices of those edge-level matrices. In performance tests we compare these stabilization parameters with the ones defined based on the element-level matrices. In both cases the formulation includes a shock-capturing term. We investigate the difference between updating the stabilization and shock-capturing parameters at the end of every time step and at the end of every nonlinear iteration within a time step. The formulation also involves activating an algorithmic feature that is based on freezing the shock-capturing parameter at its current value when a convergence stagnation is detected.*

Keywords: *Euler equations, compressible flow, finite elements, edge-based data structures, stabilization parameter*

1. Introduction

In finite element flow computations, stabilized formulations such as the streamline-upwind/Petrov-Galerkin (SUPG) Hughes and Brooks, 1979; Tezduyar and Hughes, 1982; Tezduyar and Hughes, 1983 and pressure-stabilizing/Petrov-Galerkin (PSPG) Tezduyar, 1991 formulations are widely used because of their well-pronounced advantages. The SUPG formulation for incompressible flows was first introduced in Hughes and Brooks, 1979. The SUPG formulation for compressible flows was first introduced, in the context of conservation variables, in Tezduyar and Hughes, 1982; Tezduyar and Hughes, 1983. After that, several SUPG-like methods for compressible flows were developed. Taylor–Galerkin method Donea, 1984, for example, is very similar, and under certain conditions is identical, to one of the SUPG methods introduced in Tezduyar and Hughes, 1982; Tezduyar and Hughes, 1983. Another example of the subsequent SUPG-like methods for compressible flows in conservation variables is the streamline-diffusion method described in Johnson et al., 1984. Later, following Tezduyar and Hughes, 1982; Tezduyar and Hughes, 1983, the SUPG formulation for compressible flows was recast in entropy variables and supplemented with a shock-capturing term Hughes et al., 1987. It was shown in LeBeau and Tezduyar, 1991 that the SUPG formulation introduced in Tezduyar and Hughes, 1982; Tezduyar and Hughes, 1983, when supplemented with a similar shock-capturing term, is very comparable in accuracy to the one that was recast in entropy variables. The PSPG formulation for the Navier–Stokes equations of incompressible flows, introduced in Tezduyar, 1991, assures numerical stability while allowing us to use equal-order interpolation functions for velocity and pressure. An earlier version of this stabilized formulation for Stokes flow was reported in Hughes et al., 1986.

A stabilization parameter that is almost always known as "τ" is embedded in the SUPG and PSPG formulations. This parameter involves a measure of the local length scale (also known as "element length") and other parameters such as the local Reynolds and Courant numbers. Various element lengths and τs were proposed starting with those in Hughes and Brooks, 1979 and Tezduyar and Hughes, 1982; Tezduyar and Hughes, 1983, followed by the one introduced in Tezduyar and Park, 1986, and those proposed in the subsequently reported SUPG and PSPG methods. Here we will call the SUPG formulation introduced in Tezduyar and Hughes, 1982; Tezduyar and Hughes, 1983 for compressible flows "$(SUPG)_{82}$", and the set of τs introduced in conjunction

with that formulation "τ_{82}". The stabilized formulation introduced in Tezduyar and Park, 1986 for advection–diffusion–reaction equations included a shock-capturing term and a τ definition that takes into account the interaction between the shock-capturing and SUPG terms. That τ definition precludes "compounding" (i.e. augmentation of the SUPG effect by the shock-capturing effect when the advection and shock-capturing directions coincide). The τ used in LeBeau and Tezduyar, 1991 with $(SUPG)_{82}$ is a slightly modified version of τ_{82}. A shock-capturing parameter, which we will call here "δ_{91}", was embedded in the shock-capturing term used in LeBeau and Tezduyar, 1991. Subsequent minor modifications of τ_{82} took into account the interaction between the shock-capturing and the $(SUPG)_{82}$ terms in a fashion similar to how it was done in Tezduyar and Park, 1986 for advection–diffusion–reaction equations. All these slightly modified versions of τ_{82} have always been used with the same δ_{91}, and we will categorize them here all under the label "$\tau_{82\text{-MOD}}$".

Recently, new ways of computing the τs based on the element-level matrices and vectors were introduced in Tezduyar and Osawa, 2000 in the context of the advection–diffusion equation and the Navier–Stokes equations of incompressible flows. These new definitions are expressed in terms of the ratios of the norms of the relevant matrices or vectors. They automatically take into account the local length scales, advection field and the element-level Reynolds number. Based on these definitions, a τ can be calculated for each element, or even for each element node or degree of freedom or element equation. It was pointed out in Tezduyar and Osawa, 2000; Tezduyar, 2001 that the τs to be used in advancing the solution from time level n to $n+1$ (including the τ embedded in the "LSIC stabilization" term, which resembles a discontinuity-capturing term) should be evaluated at time level n (i.e. based on the flow field already computed for time level n), so that we are spared from another level of nonlinearity.

In Catabriga et al., 2002, the τ definitions based on the element-level matrices were applied to the $(SUPG)_{82}$ formulation for inviscid compressible flows supplemented with the shock-capturing term involving δ_{91}. These concepts are extended in this paper to an edge-based implementation that was introduced in Catabriga and Coutinho, 2002. We investigate the performance differences between updating the stabilization and shock-capturing parameters at the end of every time step and at the end of every nonlinear iteration within a time step. The formulation includes activating an algorithmic feature, which was introduced earlier and is based on freezing the shock-capturing parameter at its current value when a convergence stagnation is detected.

2. Euler Equations

The system of conservation laws governing inviscid, compressible fluid flow are the Euler equations. These equations, restricted to two spatial dimensions, may be written in terms of conservation variables $\mathbf{U} = (\rho, \rho u, \rho v, \rho e)$, as

$$\mathbf{U}_{,t} + \mathbf{F}_{x,x} + \mathbf{F}_{y,y} = \mathbf{0} \quad \text{on} \quad \Omega \times [0,T] \tag{1}$$

where $\mathbf{F}_x$ and $\mathbf{F}_y$ are the Euler fluxes given elsewhere Hirsh, 1992, Ω is a domain in $\mathbb{R}^2$ and T is a positive real number. We denote the spatial and temporal coordinates respectively by $\mathbf{x} = (x,y) \in \overline{\Omega}$ and $t \in [0,T]$, where the superimposed bar indicates set closure, and Γ is the boundary of domain Ω. Here ρ is the fluid density; $\mathbf{u} = (u_x, u_y)^T$ is the velocity vector; e is the total energy per unit mass. We add to Eq. (1) the ideal gas assumption, relating pressure with the total energy per unit mass and kinetic energy. Alternatively, Eq. (1) may be written as,

$$\mathbf{U}_{,t} + \mathbf{A}_x \mathbf{U}_{,x} + \mathbf{A}_y \mathbf{U}_{,y} = \mathbf{0} \quad \text{on} \quad \Omega \times [0,T] \tag{2}$$

where $\mathbf{A}_i = \frac{\partial \mathbf{F}_i}{\partial \mathbf{U}}$. Associated to Eq. (2) we have proper boundary and initial conditions.

3. Stabilized formulation and stabilization parameters

Considering a standard discretization of Ω into finite elements, the $(SUPG)_{82}$ formulation for the Euler equations in conservation variables introduced by Tezduyar and Hughes, 1982 and Tezduyar and Hughes, 1983 is written as,

$$\begin{aligned} \int_\Omega \mathbf{W}^h . \left(\frac{\partial \mathbf{U}^h}{\partial t} + \mathbf{A}_i^h \frac{\partial \mathbf{U}^h}{\partial x_i} \right) d\Omega \quad + \\ \sum_{e=1}^{n_{el}} \int_{\Omega^e} (\mathbf{A}_k^h)^t \boldsymbol{\tau} \left(\frac{\partial \mathbf{W}^h}{\partial x_k} \right) . \left[\frac{\partial \mathbf{U}^h}{\partial t} + \mathbf{A}_i^h \frac{\partial \mathbf{U}^h}{\partial x_i} \right] d\Omega + \sum_{e=1}^{n_{el}} \int_{\Omega^e} \delta_{91} \frac{\partial \mathbf{W}^h}{\partial x_i} . \frac{\partial \mathbf{U}^h}{\partial x_i} d\Omega \quad = \quad 0 \end{aligned} \tag{3}$$

where $\mathbf{W}^h$ and $\mathbf{U}^h$, respectively the discrete weighting and test functions, are defined on standard finite element spaces. In (3) the first integral corresponds to the Galerkin formulation, the first series of element-level integrals are the SUPG stabilization terms, and the second series of element-level integrals are the shock-capturing terms added to the variational formulation to prevent spurious oscillations around shocks. The shock-capturing parameter, δ_{91}, is evaluated here using the approach proposed by LeBeau and Tezduyar, 1991. We define the following element-level matrices:

$$\mathbf{m} : \int_{\Omega^e} \mathbf{W}^h \frac{\partial \mathbf{U}^h}{\partial t} d\Omega \tag{4}$$

$$\tilde{\mathbf{c}} : \int_{\Omega^e} \left(\frac{\partial \mathbf{W}^h}{\partial x} . \mathbf{A}_x^h \frac{\partial \mathbf{U}^h}{\partial t} + \frac{\partial \mathbf{W}^h}{\partial y} . \mathbf{A}_y^h \frac{\partial \mathbf{U}^h}{\partial t} \right) d\Omega \tag{5}$$

$$\mathbf{c} : \int_{\Omega^e} \left(\mathbf{W}^h . \mathbf{A}_x^h \frac{\partial \mathbf{U}^h}{\partial x} + \mathbf{W}^h . \mathbf{A}_y^h \frac{\partial \mathbf{U}^h}{\partial y} \right) d\Omega \tag{6}$$

$$\tilde{\mathbf{k}} : \int_{\Omega^e} \left(\frac{\partial \mathbf{W}^h}{\partial x} . \mathbf{A}_x^h \mathbf{A}_x^h \frac{\partial \mathbf{U}^h}{\partial x} + \frac{\partial \mathbf{W}^h}{\partial x} . \mathbf{A}_x^h \mathbf{A}_y^h \frac{\partial \mathbf{U}^h}{\partial y} + \frac{\partial \mathbf{W}^h}{\partial y} . \mathbf{A}_y^h \mathbf{A}_x^h \frac{\partial \mathbf{U}^h}{\partial x} + \frac{\partial \mathbf{W}^h}{\partial y} . \mathbf{A}_y^h \mathbf{A}_y^h \frac{\partial \mathbf{U}^h}{\partial y} \right) d\Omega \tag{7}$$

The conventional finite element data structure associates to each triangle e its connectivity, that is, the mesh nodes I, J and K. In the edge-based data structure each edge s is associated to the adjacent elements e and f, thus to the nodes I, J, K and L, as shown in Fig. (1). Moreover, each element matrix can be disassembled into its contributions to three edges, s, $s+1$ and $s+2$, with connectivities IJ, JK and KI, that is,

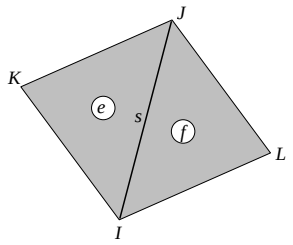


Figure 1: Elements adjacent to edge s, formed by nodes I e J.

$$\underbrace{\begin{bmatrix} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \end{bmatrix}}_{\text{element } e} = \underbrace{\begin{bmatrix} \times & \times & \mathbf{0} \\ \times & \times & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}}_{\text{edge } s} + \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \times & \times \\ \mathbf{0} & \times & \times \end{bmatrix}}_{\text{edge } s+1} + \underbrace{\begin{bmatrix} \times & \mathbf{0} & \times \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \times & \mathbf{0} & \times \end{bmatrix}}_{\text{edge } s+2} \tag{8}$$

where $\bullet$ and $\times$ are 4×4 submatrices and $\mathbf{0}$ represents a 4×4 null matrix. Thus, all the contributions pertaining to edge s will be present in the adjacent elements e and f. The resulting edge matrix is the sum of the corresponding sub-element matrices containing all the contributions to nodes I and J, that is,

$$\underbrace{\begin{bmatrix} \circ & \circ \\ \circ & \circ \end{bmatrix}}_{\text{edge } s} = \underbrace{\begin{bmatrix} \times & \times \\ \times & \times \end{bmatrix}}_{\text{element } e} + \underbrace{\begin{bmatrix} \times & \times \\ \times & \times \end{bmatrix}}_{\text{element } f} \tag{9}$$

where $\circ$ and $\times$ also represent 4×4 blocks. Considering a conventional elementwise description of a given finite element mesh, the topological informations are manipulated, generating a new edge-based mesh description. We may arrive to the edge-based matrices by

$$\mathbf{m} = \frac{(A^e + A^f)}{12} \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{I} \end{bmatrix} \quad \tilde{\mathbf{c}} = \begin{bmatrix} -\mathbf{sm}^1 & \mathbf{sm}^2 \\ \mathbf{sm}^1 & -\mathbf{sm}^2 \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} -\mathbf{scg}^1 & \mathbf{scg}^1 \\ \mathbf{scg}^2 & -\mathbf{scg}^2 \end{bmatrix} \quad \tilde{\mathbf{k}} = \begin{bmatrix} -\mathbf{scpg}^1 & \mathbf{scpg}^1 \\ \mathbf{scpg}^2 & -\mathbf{scpg}^2 \end{bmatrix} \tag{10}$$

The submatrices defined above are,

$$\mathbf{sm}^1 = \mathbf{scg}^1 = \frac{1}{6} \left[(y_{KI} + y_{IL}) \mathbf{A}_x^h + (x_{IK} + x_{LI}) \mathbf{A}_y^h \right] \tag{11}$$

$$\mathbf{sm}^2 = \mathbf{scg}^2 = \frac{1}{6} \left[(y_{JK} + y_{LJ}) \mathbf{A}_x^h + (x_{KJ} + x_{JL}) \mathbf{A}_y^h \right] \tag{12}$$

$$\mathbf{scpg}^1 = s_1 \mathbf{A}_x^h \mathbf{A}_x^h + s_2 \mathbf{A}_x^h \mathbf{A}_y^h + s_3 \mathbf{A}_y^h \mathbf{A}_x^h + s_4 \mathbf{A}_y^h \mathbf{A}_y^h \tag{13}$$

$$\mathbf{scpg}^2 = s_1 \mathbf{A}_x^h \mathbf{A}_x^h + s_3 \mathbf{A}_x^h \mathbf{A}_y^h + s_2 \mathbf{A}_y^h \mathbf{A}_x^h + s_4 \mathbf{A}_y^h \mathbf{A}_y^h \tag{14}$$

where

$$s_1 = \frac{1}{4A^e} y_{JK} y_{KJ} + \frac{1}{4A^f} y_{LJ} y_{IL} \qquad s_2 = \tfrac{1}{4A^e} y_{JK} x_{IL} + \tfrac{1}{4A^f} y_{LJ} x_{LI}$$
$$s_3 = \frac{1}{4A^e} x_{KJ} y_{KI} + \frac{1}{4A^f} x_{JL} y_{IL} \qquad s_4 = \tfrac{1}{4A^e} x_{KJ} x_{IK} + \tfrac{1}{4A^f} x_{JL} x_{LI} \tag{15}$$

In the equations above, A^e and A^f are, respectivaly, area of the e and f adjacents elements, $\mathbf{I}$ is the identity matrix of order 4 and $x_{ij} = x_i - x_j$, $y_{ij} = y_i - y_j$, $i, j = I, J, K, L$.

3.1. Computing τ by edge matrix norms

We define the SUPG parameters from the edge matrices as given in Tezduyar and Osawa, 2000,

$$\tau_{S1} = \frac{\|\mathbf{c}\|}{\|\tilde{\mathbf{k}}\|} \qquad \tau_{S2} = \frac{\Delta t}{2} \frac{\|\mathbf{c}\|}{\|\tilde{\mathbf{c}}\|} \qquad \tau_{SUPG} = \left(\frac{1}{\tau_{S1}^r} + \frac{1}{\tau_{S2}^r} \right)^{-1/r} \tag{16}$$

where Δt is the time step, $\|\mathbf{b}\| = max_{1 \le j \le n_{ee}} \{ |b_{1j}| + |b_{2j}| + \ldots + |b_{n_{ee},j}| \}$, n_{ee} is the number of edge equations, that is, the number of edge nodes times the number of degrees of freedom per node and r is an integer parameter.

3.2. Computing τ by dof edge submatrix norms

We can calculate a separate τ for each edge matrix degree of freedom. The resulting diagonal stabilization tensor is,

$$\boldsymbol{\tau}_{SUPG} = \begin{bmatrix} \tau_\rho & & & \\ & \tau_u & & \\ & & \tau_v & \\ & & & \tau_e \end{bmatrix} \tag{17}$$

where the subindexes (ρ, u, v, e) are the primitive variables. Each τ_i can be to calculate by

$$\tau_i = \left(\frac{1}{\tau_{S1_i}^r} + \frac{1}{\tau_{S2_i}^r} \right)^{-1/r} \qquad \tau_{S1_i} = \frac{\|\mathbf{c}_i\|}{\|\tilde{\mathbf{k}}_i\|} \qquad \tau_{S2_i} = \frac{\Delta t}{2} \frac{\|\mathbf{c}_i\|}{\|\tilde{\mathbf{c}}_i\|} \tag{18}$$

where Δt is the time step. $\mathbf{c}_i$, $\tilde{\mathbf{k}}_i$ and $\tilde{\mathbf{c}}_i$ are the submatrices of the edge matrices for each degree of freedom $i = \rho, u, v, e$. Theses submatrices are defined by edge matrix coefficients. Lets a general degree of freedom be denoted i and the corresponding edge submatrix $\mathbf{b}_i$ can be given by,

$$\mathbf{b}_i = \begin{bmatrix} b_{p_1,1} & b_{p_1,2} & \ldots & b_{p_1,8} \\ b_{p_2,1} & b_{p_2,2} & \ldots & b_{p_2,8} \end{bmatrix} \qquad \text{where} \quad \begin{array}{llll} \text{if} & i = \rho & \text{then} & (p_1, p_2) \equiv (1, 5) \\ \text{if} & i = u & \text{then} & (p_1, p_2) \equiv (2, 6) \\ \text{if} & i = v & \text{then} & (p_1, p_2) \equiv (3, 7) \\ \text{if} & i = e & \text{then} & (p_1, p_2) \equiv (4, 8) \end{array} \tag{19}$$

The norms of these submatrices, used in Eq. (18), are computed by $\|\mathbf{b}_i\| = max_{1 \le j \le n_{ee}} \{ |b_{p_1,j}| + |b_{p_2,j}| \}$.

4. Numerical Results

In this section we compare results obtained using the element and edge data structures. Tolerance of preconditioned GMRES algorithm is set to 0.1, the dimension of the Krylov subspace to 5 and the number of multicorrections fixed to 3. All the solutions are initialized with free-stream values. The symbol τ_g represents τs calculated based on element-level and edge-level matrices. The symbol τ_{dof} represents τs calculated based on the degree-of-freedom sub-matrices of the element-level and edge-level matrices.

4.1. Oblique Shock

The first problem consists of a two-dimensional steady problem of a inviscid, Mach 2, uniform flow, over a wedge at an angle of -10° with respect to a horizontal wall, resulting in the occurrence of an oblique shock with an angle of 29.3° emanating from the leading edge of the wedge, as shown in Fig. (2).

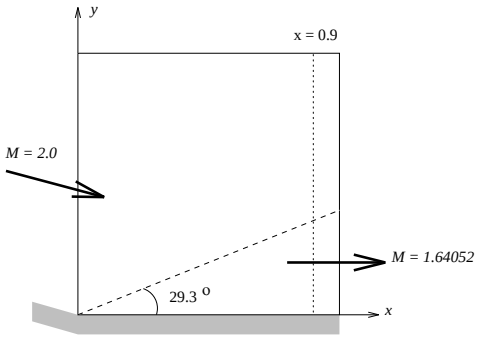


Figure 2: Oblique shock - problem description.

The computational domain is the square $0 \leq x \leq 1$ and $0 \leq y \leq 1$. Prescribing the following flow data at the inflow, i.e., on the left and top sides of the shock, results in the exact solution with the flow data past the shock:

$$\text{Inflow} \begin{cases} M &= 2.0 \\ \rho &= 1.0 \\ u_1 &= cos10^0 \\ u_2 &= -sin10^0 \\ p &= 0.17857 \end{cases} \qquad \text{Outflow} \begin{cases} M &= 1.64052 \\ \rho &= 1.45843 \\ u_1 &= 0.88731 \\ u_2 &= 0.0 \\ p &= 0.30475 \end{cases} \tag{20}$$

where M is the Mach number, ρ is the flow density, u_1 and u_2 are the horizontal and vertical velocities respectively, and p is the pressure.

Four Dirichlet boundary conditions are imposed on the left and top boundaries; the slip condition $u_2 = 0$ is set at the bottom boundary; and no boundary conditions are imposed on the outflow (right) boundary. A 20×20 mesh with 800 linear triangles and 441 nodes is employed.

Table 1: Oblique shock - Computational costs (in number of time steps, GMRES iterations and CPU seconds) for iteration update and time-step update (both with freezing shock-capturing parameter) - using τ_g and τ_{dof} - element-level and edge-level.

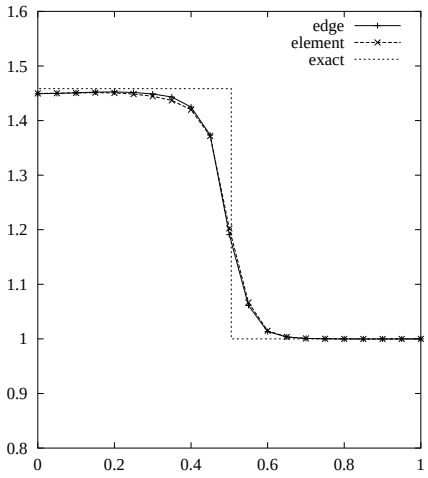
Iteration update						
Data	τ_g			τ_{dof}		
structure	N_{GMRES}	N_{steps}	$time(sec)$	N_{GMRES}	N_{steps}	$time(sec)$
Edge	4,162	797	84	4,088	787	88
Element	4,287	846	149	4,902	833	97
Time-step update						
Data	τ_g			τ_{dof}		
structure	N_{GMRES}	N_{steps}	$time(sec)$	N_{GMRES}	N_{steps}	$time(sec)$
Edge	4,845	833	85	4,965	822	87
Element	4,246	838	145	4,857	819	93

Figures (3-4) and Table (1) show the performance of τ_g and τ_{dof} based on both the element-level and edge-level approaches.

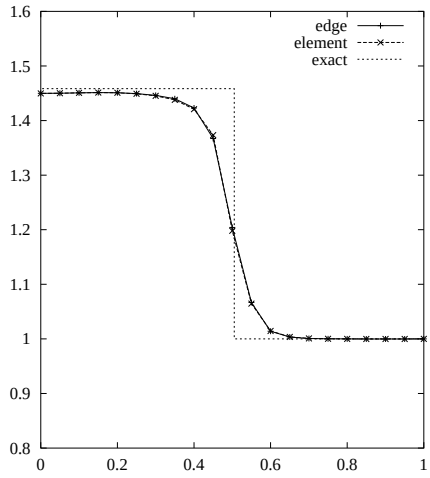
4.2. Reflected Shock

This two-dimensional steady problem consists of three regions (R1, R2 and R3) separated by an oblique shock and its reflection from a wall, as shown in Fig. (5). Prescribing the following Mach 2.9 flow data at the inflow, i.e., the first region on the left (R1), and requiring that the incident shock to be at an angle of 29°, leads to the exact solution (R2 and R3):

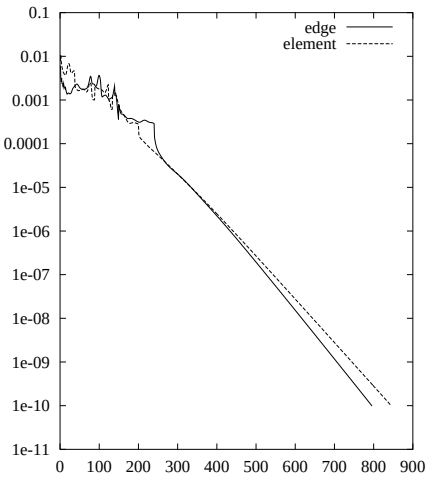
$$\text{R1} \begin{cases} M &= 2.9 \\ \rho &= 1.0 \\ u_1 &= 2.9 \\ u_2 &= 0.0 \\ p &= 0.714286 \end{cases} \quad \text{R2} \begin{cases} M &= 2.3781 \\ \rho &= 1.7 \\ u_1 &= 2.61934 \\ u_2 &= -0.50632 \\ p &= 1.52819 \end{cases} \quad \text{R3} \begin{cases} M &= 1.94235 \\ \rho &= 2.68728 \\ u_1 &= 2.40140 \\ u_2 &= 0.0 \\ p &= 2.93407 \end{cases} \tag{21}$$



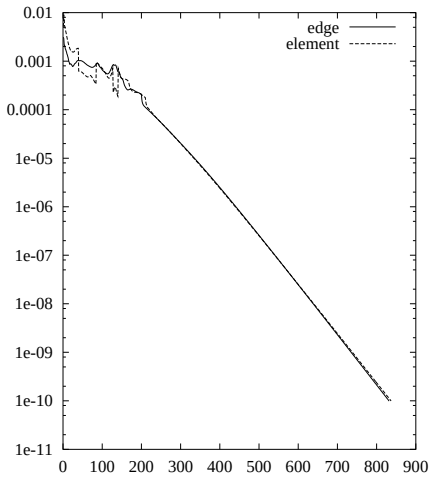
(a) Density profile at $x = 0.9$ - iteration update.



(b) Density profile at $x = 0.9$ - time-step update.

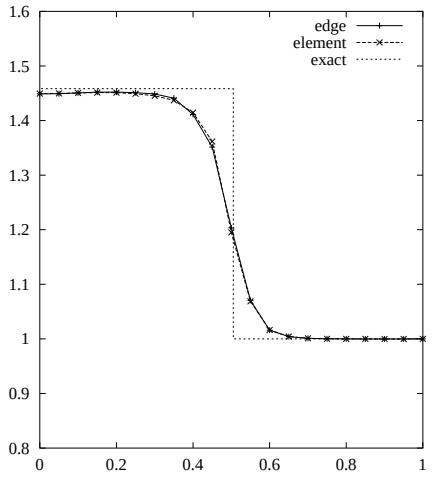


(c) Evolution of density residual - iteration update.

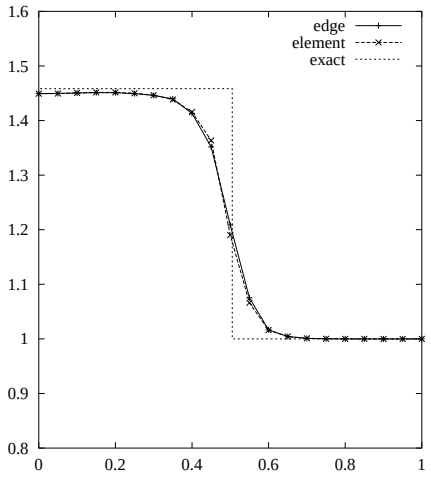


(d) Evolution of density residual - time-step update.

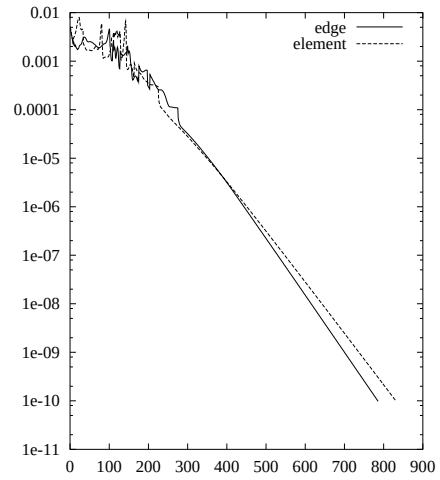
Figure 3: Oblique shock - solutions and residual with iteration update and time-step update (both with freezing shock-capturing parameter) - using τ_g - element-level and edge-level.



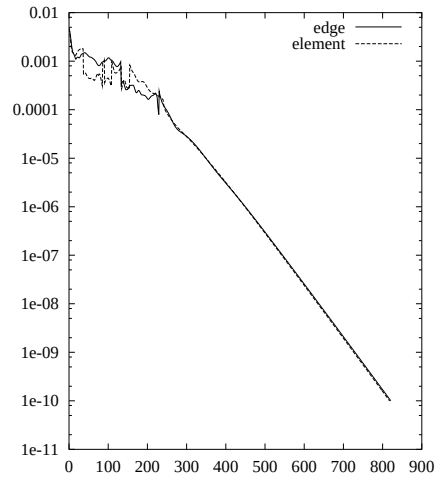
(a) Density profile at $x = 0.9$ - iteration update.



(b) Density profile at $x = 0.9$ - time-step update.



(c) Evolution of density residual - iteration update.



(d) Evolution of density residual - time-step update.

Figure 4: Oblique shock - solutions and residual with iteration update and time-step update (both with freezing shock-capturing parameter) - using τ_{dof} - element-level and edge-level.

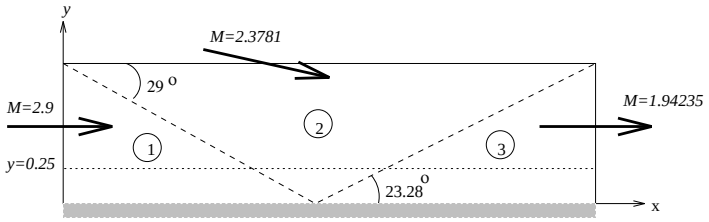
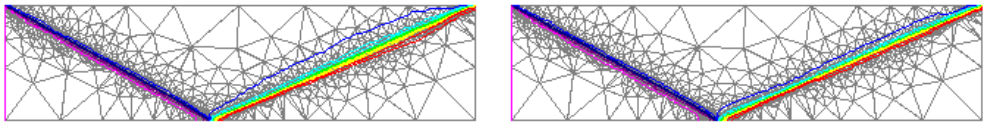


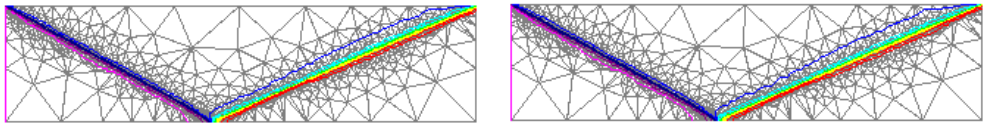
Figure 5: Reflected shock - problem description.

We prescribe density, velocities and pressure on the left and top boundaries; the slip condition is imposed on the wall (bottom boundary); and no boundary conditions are set on the outflow (right) boundary. We consider an unstructured mesh with 1,837 nodes and 3,429 elements covering the domain $0 \leq x \leq 4.1$ and $0 \leq y \leq 1$.



(a) Density contours with τ_g parameter with iteration update.

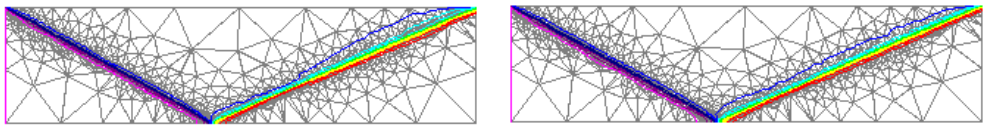
(b) Density contours with τ_g parameter with time-step update.



(c) Density contours with τ_{dof} parameter with iteration update.

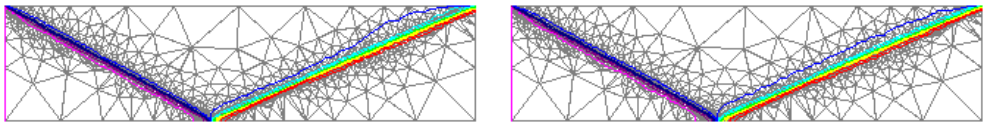
(d) Density contours with τ_{dof} parameter with time-step update.

Figure 6: Reflected shock - density computed with iteration update and time-step update (both with freezing shock-capturing parameter) - using τ_g and τ_{dof} - edge-level.



(a) Density contours with τ_g parameter with iteration update.

(b) Density contours with τ_g parameter with time-step update.



(c) Density contours with τ_{dof} parameter with iteration update.

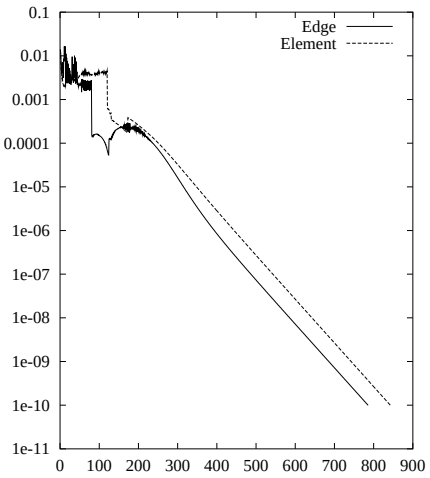
(d) Density contours with τ_{dof} parameter with time-step update.

Figure 7: Reflected shock - density computed with iteration update and time-step update (both with freezing shock-capturing parameter) - using τ_g and τ_{dof} - element-level.

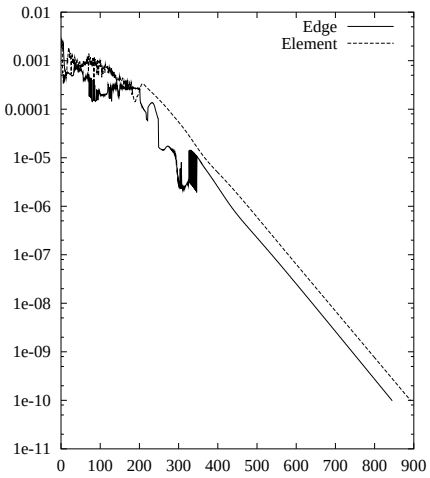
Figures (6-9) and Table (2) show the performance of τ_g and τ_{dof} based on both the element-level and edge-level approaches.

5. Concluding remarks

We highlighted, for the SUPG formulation of inviscid compressible flows, stabilization parameters defined based the edge-level matrices and the degree-of-freedom submatrices of those edge-level matrices. These definitions are expressed in terms of the ratios of the norms of the relevant matrices, and take automatically into account the flow field, the local length scales, and the time step size. By inspecting the solution quality and convergence history, we compared these stabilization parameters with the ones defined based on the element-level matrices. In both cases the formulation includes a shock-capturing parameter. Also by inspecting the solution quality and convergence history, we investigated the performance difference between updating the stabilization and shock-capturing parameters at the end of every time step and at the end of every nonlinear

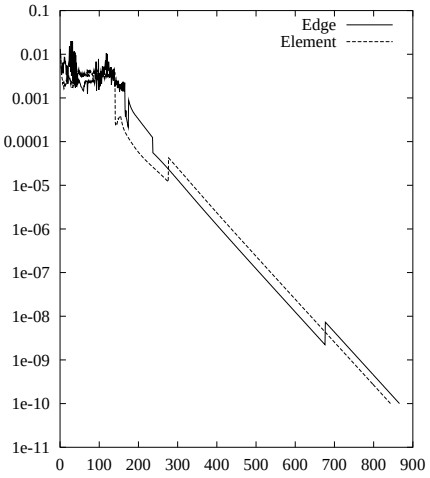


(a) Evolution of density residual - iteration update.

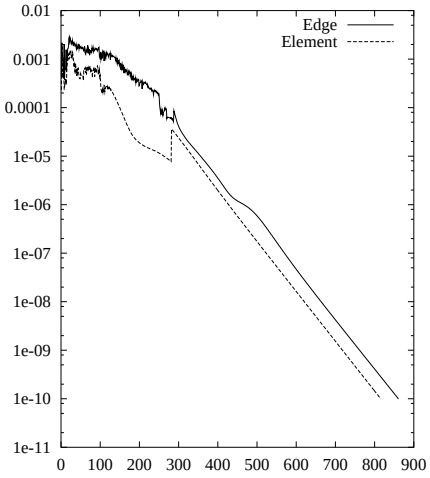


(b) Evolution of density residual - time-step update.

Figure 8: Reflected shock - residuals with iteration update and time-step update (both with freezing shock-capturing parameter) - using τ_g - element-level and edge-level.



(a) Evolution of density residual - iteration update



(b) Evolution of density residual - time-step update

Figure 9: Reflected shock - residuals with iteration update and time-step update (both with freezing shock-capturing parameter) - using τ_{dof} - element-level and edge-level.

Table 2: Reflected shock - Computational costs (in number of time steps, GMRES iterations and CPU seconds) for iteration update and time-step update (both with freezing shock-capturing parameter) - using τ_g and τ_{dof} - element-level and edge-level.

Iteration update						
Data structure	τ_g			τ_{dof}		
	N_{GMRES}	N_{steps}	$time(sec)$	N_{GMRES}	N_{steps}	$time(sec)$
Edge	10,237	787	500	9,689	867	544
Element	9,127	844	948	9,138	845	584
Time-step update						
Data structure	τ_g			τ_{dof}		
	N_{GMRES}	N_{steps}	$time(sec)$	N_{GMRES}	N_{steps}	$time(sec)$
Edge	11,161	846	544	9,133	862	528
Element	9,913	893	984	9,278	817	579

iteration within a time step. The formulation also involves activating an algorithmic feature that is based on freezing the shock-capturing parameter at its current value when a convergence stagnation is detected. We observe that in all cases the solution qualities are very comparable. In terms of computational efficiency, τ definitions based on the element-level matrices are responding better to using degree-of-freedom submatrices (in place of full matrices) than the definitions based on the edge-level matrices. As it was observed earlier, the edge-based implementation is computationally more efficient than the element-based implementation. However, this advantage is less pronounced for the degree-of-freedom submatrices than it is for the full matrices.

6. Acknowledgments

This work is partly supported by grant CNPq/PROTEM-CC 68.0066/01-2 and CNPq/MCT 522692/95-8. The third author was supported by the US Army Natick Soldier Center and NASA Johnson Space Center.

7. References

Catabriga, L., Coutinho, A., and Tezduyar, T. E., 2002, Finite element SUPG parameters computed from local Matrices for compressible flows, "Proceedings of the 9th Brazilian Congress of Thermal Engineering and Sciences CD-ROM", pp. 1–11, Caxambu, Minas Gerais, Brazil.

Catabriga, L. and Coutinho, A. L. G. A., 2002, Implicit SUPG solution of Euler equations using edge-based data structures, "Comput. Methods Appl. Mech. and Engrg.", Vol. 191, pp. 3477–3490.

Donea, J., 1984, A Taylor-Galerkin method for convective transport problems, "International Journal for Numerical Methods in Engineering", Vol. 20, pp. 101–120.

Hirsh, C., 1992, "Numerical Computation of Internal and External Flows - Computational Methods for Inviscid and Viscous Flows, Vol. 2", John Wiley and Sons Ltd, Chichester.

Hughes, T. J. R. and Brooks, A. N., 1979, A multi-dimensional upwind scheme with no crosswind diffusion, Hughes, T., editor, "Finite Element Methods for Convection Dominated Flows", Vol. 34, pp. 19–35, New York. ASME.

Hughes, T. J. R., Franca, L. P., and Balestra, M., 1986, A new finite element formulation for computational fluid dynamics: V. Circumventing the Babuška–Brezzi condition: A stable Petrov–Galerkin formulation of the Stokes problem accommodating equal-order interpolations, "Computer Methods in Applied Mechanics and Engineering", Vol. 59, pp. 85–99.

Hughes, T. J. R., Franca, L. P., and Mallet, M., 1987, A new finite element formulation for computational fluid dynamics: VI. Convergence analysis of the generalized SUPG formulation for linear time-dependent multi-dimensional advective-diffusive systems, "Computer Methods in Applied Mechanics and Engineering", Vol. 63, pp. 97–112.

Johnson, C., Navert, U., and Pitkaranta, J., 1984, Finite element methods for linear hyperbolic problems, "Computer Methods in Applied Mechanics and Engineering", Vol. 45, pp. 285–312.

LeBeau, G. J. and Tezduyar, T. E., 1991, Finite element computation of compressible flows with the SUPG formulation, Dhaubhadel, M., Engelman, M., and Reddy, J., editors, "Advances in Finite Element Analysis in Fluid Dynamics", Vol. 123, pp. 21–27, New York. ASME.

Tezduyar, T. and Park, Y., 1986, Discontinuity capturing finite element formulations for nonlinear convection-diffusion-reaction problems, "Computer Methods in Applied Mechanics and Engineering", Vol. 59, pp. 307–325.

Tezduyar, T. E., 1991, Stabilized finite element formulations for incompressible flow computations, "Advances in Applied Mechanics", Vol. 28, pp. 1–44.

Tezduyar, T. E., 2001, Adaptive determination of the finite element stabilization parameters, "European Congress on Computational Methods in Applied Sciences and Engineering-ECCOMAS Computational Fluid Dynamics Conference CD-ROM", pp. 1–13, Swansea, Wales, UK.

Tezduyar, T. E. and Hughes, T. J. R., 1982, Development of Time-accurate Finite Element Techniques for First-order Hyperbloic Systems with Particular Emphasis on the Compressible Euler Equations, NASA Technical Report NASA-CR-204772, NASA.

Tezduyar, T. E. and Hughes, T. J. R., 1983, Finite element formulations for convection dominated flows with particular emphasis on the compressible Euler equations, "21st Aerospace Sciences Meeting", Vol. 83-0125, Reno, Nevada. AIAA.

Tezduyar, T. E. and Osawa, Y., 2000, Finite element stabilization parameters computed from element matrices and vectors, "Computer Methods in Applied Mechanics and Engineering", Vol. 190, pp. 411–430.