Comput. Mech. (2006) 38: 334–343
DOI 10.1007/s00466-006-0033-1

ORIGINAL PAPER

Lucia Catabriga · Alvaro L. G. A. Coutinho
Tayfun E. Tezduyar

Compressible flow SUPG stabilization parameters computed from degree-of-freedom submatrices

Received: 1 August 2005 / Accepted: 23 December 2005 / Published online: 9 February 2006

Abstract We present, for the SUPG formulation of inviscid compressible flows, stabilization parameters defined based on the degree-of-freedom submatrices of the element-level matrices. With 2D steady-state test problems involving supersonic flows and shocks, we compare these stabilization parameters with the ones defined based on the full element-level matrices. We also compare them to the stabilization parameters introduced in the earlier development stages of the SUPG formulation of compressible flows. In all cases the formulation includes a shock-capturing term involving a shock-capturing parameter. 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 includes, as an option, an algorithmic feature that is based on freezing the shock-capturing parameter at its current value when a convergence stagnation is detected.

Keywords Inviscid compressible flows · Finite elements · SUPG formulation · Stabilization parameters · Degree-of-freedom submatrices

L. Catabriga
Department of Computer Science,
Federal University of Espírito Santo (UFES),
Av. Fernando Ferrari, 514, Goiabeiras,
29075-910, Vitória, Brazil
E-mail: luciac@inf.ufes.br

A. L. G. A. Coutinho (✉)
Department of Civil Engineering - COPPE,
Federal University of Rio de Janeiro (UFRJ),
Caixa Postal 68506, Rio de Janeiro, Brazil
E-mail: alvaro@nacad.ufrj.br

T. E. Tezduyar
Team for Advanced Flow Simulation
and Modeling (T*AFSM), Mechanical Engineering,
Rice University - MS 321,
6100 Main Street, Houston, TX 77005, USA
E-mail: tezduyar@rice.edu

1 Introduction

The streamline-upwind/Petrov-Galerkin (SUPG) formulation of compressible flows [11, 19, 20] is widely used in finite element flow computations. In a more general framework, stabilized formulations such as the SUPG formulation of compressible flows, SUPG formulation of incompressible flows [1, 10], and the pressure-stabilizing/Petrov-Galerkin (PSPG) formulation [15] have become very popular because of their well-known desirable features. These formulations prevent numerical instabilities in solving problems with high Reynolds or Mach numbers and shocks or thin boundary layers, as well as when using equal-order interpolation functions for velocity and pressure. The SUPG and PSPG formulations achieve these objectives without introducing excessive numerical dissipation. Furthermore, as it was pointed out in [23], these stabilized formulations also substantially improve the convergence rate in iterative solution of the large matrix systems that need to be solved at every Newton–Raphson step.

The SUPG formulation for incompressible flows was first introduced in an ASME paper [10]. Additional studies and examples were presented in a journal paper [1]. The SUPG formulation for compressible flows was first introduced, in the context of conservation variables, in a NASA technical report [19]. A concise version of that was published as an AIAA paper [20], and a more comprehensive version as a journal paper [11]. Several SUPG-like methods for compressible flows were developed after that. For example, Taylor–Galerkin method [9] is very similar, and under certain conditions is identical, to one of the SUPG methods introduced in [11, 19, 20]. Another example of the subsequent SUPG-like methods for compressible flows in conservation variables is the streamline-diffusion method described in [13]. Later, following [19, 20, 11], the SUPG formulation for compressible flows was recast in entropy variables and supplemented with a shock-capturing term [12]. It was shown in [14] that the SUPG formulation introduced in [11, 19, 20], when supplemented with a similar shock-capturing term, is

very comparable in accuracy to the one that was recast in entropy variables.

A stabilization parameter that is mostly known as "τ" is embedded in the SUPG and PSPG formulations. It involves a measure of the local length scale (also known as "element length") and other parameters such as the element Reynolds and Courant numbers. Various element lengths and τs were proposed starting with those in [1, 10, 11, 19, 20], followed by the one introduced in [22], and those proposed in the subsequently reported SUPG and PSPG methods. In this paper we will call the SUPG formulation introduced in [1, 19, 20] for compressible flows "$(SUPG)_{82}$", and the set of τs introduced in conjunction with that formulation "τ_{82}". The stabilized formulation introduced in [22] 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 directions coincide). The τ used in [14] with $(SUPG)_{82}$ is a slightly modified version of τ_{82}. A shock-capturing parameter, which we will call in this paper "δ_{91}", was embedded in the shock-capturing term used in [14]. 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 [22] 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 "$\tau_{82\text{-MOD}}$". Calculating the τs based on the element-level matrices and vectors was introduced in [21] in the context of the advection–diffusion equation and the Navier–Stokes equations of incompressible flows. These definitions are expressed in terms of the ratios of the norms of the matrices or vectors. They automatically take into account the local length scales, advection field and the element Reynolds number. Based on these definitions, a τ can be calculated for each element or for each degree-of-freedom of each element, or, as it was proposed in [17], for each integration point of each element. It was proposed in [16, 18, 21] that the stabilization parameters to be used in advancing the solution from time level n to $n+1$ (including the parameter embedded in a stabilization terms that 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). This way we are spared from another level of nonlinearity.

In [4, 8], the τ definitions based on the element matrices were used in conjunction with the $(SUPG)_{82}$ formulation supplemented with the shock-capturing term involving δ_{91}. These concepts were extended in [6, 7] to the edge-based implementation that was introduced in [2]. In this paper, τs defined for each degree-of-freedom of each element (based on the degree-of-freedom submatrices of the element-level matrices) are used with the $(SUPG)_{82}$ formulation supplemented with the shock-capturing term involving δ_{91}. We reported this effort first in a conference paper [5]. We investigate the performance differences between these τ definitions, the τ definitions based on the full element-level matrices, and $\tau_{82\text{-MOD}}$. We also investigate the performance differences between calculating the stabilization and shock-capturing parameters at time level n and at (every nonlinear iteration of) time level $n+1$. The performance comparisons are based on 2D steady-state test problems involving supersonic flows and shocks. The formulation includes, as an option, 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 flows are the Euler equations. In two dimensions these equations can be written in terms of the conservation variables, $\mathbf{U} = (\rho, \rho u, \rho v, \rho e)$, as

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

Here ρ is the fluid density, $\mathbf{u} = (u, v)$ is the velocity vector, e is the total energy per unit mass, $\mathbf{F}_x$ and $\mathbf{F}_y$ are the Euler fluxes, 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. We consider ideal gases. Equation (1) can also be written as

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

where $\mathbf{A}_x = \frac{\partial \mathbf{F}_x}{\partial \mathbf{U}}$ and $\mathbf{A}_y = \frac{\partial \mathbf{F}_y}{\partial \mathbf{U}}$. We assume that we have an appropriate set of boundary and initial conditions associated with Eq. (2).

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 in [11, 19, 20], supplemented with a shock-capturing term [14], is written as

$$\begin{aligned}
&\int_{\Omega} \mathbf{W}^h \cdot \left(\frac{\partial \mathbf{U}^h}{\partial t} + \mathbf{A}_x^h \frac{\partial \mathbf{U}^h}{\partial x} + \mathbf{A}_y^h \frac{\partial \mathbf{U}^h}{\partial y} \right) \mathrm{d}\Omega \\
&\quad + \sum_{e=1}^{n_{el}} \int_{\Omega^e} \left(\left(\boldsymbol{\tau} \frac{\partial \mathbf{W}^h}{\partial x} \right) \mathbf{A}_x^h + \left(\boldsymbol{\tau} \frac{\partial \mathbf{W}^h}{\partial y} \right) \mathbf{A}_y^h \right) \\
&\quad \cdot \left(\frac{\partial \mathbf{U}^h}{\partial t} + \mathbf{A}_x^h \frac{\partial \mathbf{U}^h}{\partial x} + \mathbf{A}_y^h \frac{\partial \mathbf{U}^h}{\partial y} \right) \mathrm{d}\Omega \\
&\quad + \sum_{e=1}^{n_{el}} \int_{\Omega^e} \delta_{91} \left(\frac{\partial \mathbf{W}^h}{\partial x} \cdot \frac{\partial \mathbf{U}^h}{\partial x} + \frac{\partial \mathbf{W}^h}{\partial y} \cdot \frac{\partial \mathbf{U}^h}{\partial y} \right) \mathrm{d}\Omega = 0.
\end{aligned} \tag{3}$$

Here $\mathbf{W}^h$ and $\mathbf{U}^h$ are the finite-dimensional test and trial functions that are defined on standard finite element spaces. In Eq. (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_{sl}, is calculated here using the approach proposed in [14]. We define the following element-level matrices:

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

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

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

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

Considering the standard finite element approximation we have:

$$\mathbf{U}^h = \mathbf{N}\mathbf{v}, \quad \mathbf{W}^h = \mathbf{N}\mathbf{c}, \quad \frac{\partial \mathbf{U}^h}{\partial t} = \mathbf{N}\mathbf{a}, \tag{8}$$

where $\mathbf{v}$ is the vector of nodal values of $\mathbf{U}$ (function of time only), $\mathbf{c}$ is a vector of arbitrary constants, $\mathbf{a}$ is the time derivate of $\mathbf{v}$ ($\mathbf{a} = \mathrm{d}\mathbf{v}/\mathrm{d}t$), and $\mathbf{N}$ is a matrix containing the shape functions. For linear triangles, the element shape functions can be represented in matrix form as

$$\mathbf{N}^e = \begin{bmatrix} N_1\mathbf{I} & N_2\mathbf{I} & N_3\mathbf{I} \end{bmatrix}, \tag{9}$$

where N_1, N_2 and N_3 are the element shape functions and $\mathbf{I}$ is the identity matrix of order 4. It is useful to define the discrete local (element) gradient operator as

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_x \\ \mathbf{B}_y \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathbf{N}}{\partial x} \\ \frac{\partial \mathbf{N}}{\partial y} \end{bmatrix} = \frac{1}{2A^e} \begin{bmatrix} y_{23}\mathbf{I} & y_{31}\mathbf{I} & y_{12}\mathbf{I} \\ x_{32}\mathbf{I} & x_{13}\mathbf{I} & x_{21}\mathbf{I} \end{bmatrix}, \tag{10}$$

where A^e is the area of the element, $x_{ab} = x_a - x_b$, and $y_{ab} = y_a - y_b$ for $a, b = 1, 2, 3$. With the expressions given by Eqs. (8)–(10), we calculate the element-level matrices as follows:

$$\mathbf{m} = \int_{\Omega e} \mathbf{N}^{\mathrm{T}}\mathbf{N}\mathrm{d}\Omega^e = \frac{A^e}{12} \begin{bmatrix} 2\mathbf{I} & \mathbf{I} & \mathbf{I} \\ \mathbf{I} & 2\mathbf{I} & \mathbf{I} \\ \mathbf{I} & \mathbf{I} & 2\mathbf{I} \end{bmatrix}, \tag{11}$$

$$\tilde{\mathbf{k}} = \int_{\Omega^e} \mathbf{B}^{\mathrm{T}} \begin{bmatrix} \mathbf{A}_x^h\mathbf{A}_x^h & \mathbf{A}_x^h\mathbf{A}_y^h \\ \mathbf{A}_y^h\mathbf{A}_x^h & \mathbf{A}_y^h\mathbf{A}_y^h \end{bmatrix} \mathbf{B}\mathrm{d}\Omega^e = \frac{1}{4A^e} \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} & \mathbf{B}_{13} \\ \mathbf{B}_{21} & \mathbf{B}_{22} & \mathbf{B}_{23} \\ \mathbf{B}_{31} & \mathbf{B}_{32} & \mathbf{B}_{33} \end{bmatrix}. \tag{12}$$

The submatrices in Eq. (12) are:

$$\begin{aligned}
\mathbf{B}_{12} &= y_{23}(y_{31}\mathbf{A}_{xx} + x_{13}\mathbf{A}_{xy}) + x_{32}(y_{31}\mathbf{A}_{yx} + x_{13}\mathbf{A}_{yy}) \\
\mathbf{B}_{13} &= y_{23}(y_{12}\mathbf{A}_{xx} + x_{21}\mathbf{A}_{xy}) + x_{32}(y_{12}\mathbf{A}_{yx} + x_{21}\mathbf{A}_{yy}) \\
\mathbf{B}_{21} &= y_{31}(y_{23}\mathbf{A}_{xx} + x_{32}\mathbf{A}_{xy}) + x_{13}(y_{23}\mathbf{A}_{yx} + x_{32}\mathbf{A}_{yy}) \\
\mathbf{B}_{23} &= y_{31}(y_{12}\mathbf{A}_{xx} + x_{21}\mathbf{A}_{xy}) + x_{13}(y_{12}\mathbf{A}_{yx} + x_{21}\mathbf{A}_{yy}) \\
\mathbf{B}_{31} &= y_{12}(y_{23}\mathbf{A}_{xx} + x_{32}\mathbf{A}_{xy}) + x_{21}(y_{23}\mathbf{A}_{yx} + x_{32}\mathbf{A}_{yy}) \\
\mathbf{B}_{32} &= y_{12}(y_{31}\mathbf{A}_{xx} + x_{13}\mathbf{A}_{xy}) + x_{21}(y_{31}\mathbf{A}_{yx} + x_{13}\mathbf{A}_{yy}) \\
\mathbf{B}_{11} &= -(\mathbf{B}_{12} + \mathbf{B}_{13}) \\
\mathbf{B}_{22} &= -(\mathbf{B}_{21} + \mathbf{B}_{23}) \\
\mathbf{B}_{33} &= -(\mathbf{B}_{31} + \mathbf{B}_{32}),
\end{aligned} \tag{13}$$

where $\mathbf{A}_{ij} = \mathbf{A}_i^h\mathbf{A}_j^h$ for $i, j = x, y$.

$$\tilde{\mathbf{c}} = \int_{\Omega e} (\mathbf{B}_x^{\mathrm{T}}\mathbf{A}_x^h\mathbf{N} + \mathbf{B}_y^{\mathrm{T}}\mathbf{A}_y^h\mathbf{N})\mathrm{d}\Omega^e = \frac{1}{6} \begin{bmatrix} \mathbf{m}^1 & \mathbf{m}^1 & \mathbf{m}^1 \\ \mathbf{m}^2 & \mathbf{m}^2 & \mathbf{m}^2 \\ \mathbf{m}^3 & \mathbf{m}^3 & \mathbf{m}^3 \end{bmatrix}, \tag{14}$$

where $\mathbf{m}^1$, $\mathbf{m}^2$ and $\mathbf{m}^3$ are given by

$$\mathbf{m}^1 = y_{23}\mathbf{A}_x^h + x_{32}\mathbf{A}_y^h \tag{15}$$

$$\mathbf{m}^2 = y_{31}\mathbf{A}_x^h + x_{13}\mathbf{A}_y^h \tag{16}$$

$$\mathbf{m}^3 = y_{12}\mathbf{A}_x^h + x_{21}\mathbf{A}_y^h \tag{17}$$

$$\mathbf{c} = \int_{\Omega e} (\mathbf{N}^{\mathrm{T}}\mathbf{A}_x^h\mathbf{B}_x + \mathbf{N}^{\mathrm{T}}\mathbf{A}_y^h\mathbf{B}_y)\mathrm{d}\Omega^e = \frac{1}{6} \begin{bmatrix} \mathbf{c}^1 & \mathbf{c}^2 & \mathbf{c}^3 \\ \mathbf{c}^1 & \mathbf{c}^2 & \mathbf{c}^3 \\ \mathbf{c}^1 & \mathbf{c}^2 & \mathbf{c}^3 \end{bmatrix}, \tag{18}$$

and the submatrices $\mathbf{c}^1$, $\mathbf{c}^2$ and $\mathbf{c}^3$ are

$$\mathbf{c}^1 = y_{23}\mathbf{A}_x^h + x_{32}\mathbf{A}_y^h \tag{19}$$

$$\mathbf{c}^2 = y_{31}\mathbf{A}_x^h + x_{13}\mathbf{A}_y^h \tag{20}$$

$$\mathbf{c}^3 = y_{12}\mathbf{A}_x^h + x_{21}\mathbf{A}_y^h. \tag{21}$$

We define the SUPG stabilization parameters from the element matrices, as proposed in [21]:

$$\tau_g = \left(\frac{1}{\tau_{S1}^r} + \frac{1}{\tau_{S2}^r} \right)^{-1/r}, \tag{22}$$

where

$$\tau_{S1} = \frac{\|\mathbf{c}\|}{\|\tilde{\mathbf{k}}\|} \quad \tau_{S2} = \frac{\Delta t}{2} \frac{\|\mathbf{c}\|}{\|\tilde{\mathbf{c}}\|}. \tag{23}$$

Here Δt is the time step, $\|\mathbf{b}\| = \max_{1 \le j \le n_{\mathrm{ee}}}\{|b_{1j}| + |b_{2j}| + \cdots + |b_{n_{\mathrm{ee}},j}|\}$, n_{ee} is the number of element equations (number of element nodes $\times$ the number of degrees-of-freedom per node), and r is an integer parameter.

It was also proposed in [21] to calculate a separate τ for each element degree-of-freedom. The resulting stabilization parameter matrix is written as

$$\boldsymbol{\tau}_{\mathrm{dof}} = \begin{bmatrix} \tau_\rho & & & \\ & \tau_u & & \\ & & \tau_v & \\ & & & \tau_e \end{bmatrix}, \tag{24}$$

where the subscripts (ρ, u, v, e) refer to the primitive variables associated with each degree-of-freedom. Each τ_I can be calculated by using the expression

$$\tau_I = \left(\frac{1}{((\tau_{S1})_I)^r} + \frac{1}{((\tau_{S2})_I)^r} \right)^{-1/r}, \tag{25}$$

where

$$(\tau_{S1})_I = \frac{\|\mathbf{c}_I\|}{\|\tilde{\mathbf{k}}_I\|} \quad (\tau_{S2})_I = \frac{\Delta t}{2} \frac{\|\mathbf{c}_I\|}{\|\tilde{\mathbf{c}}_I\|}. \tag{26}$$

Here $\mathbf{c}_I$, $\tilde{\mathbf{k}}_I$ and $\tilde{\mathbf{c}}_I$ are the submatrices of the element matrices, associated with each degree-of-freedom $I = \rho, u, v, e$. The element submatrix $\boldsymbol{b}_I$ corresponding to the degree-of-freedom I can be written as

$$\boldsymbol{b}_I = \begin{bmatrix} b_{p_1,1} & b_{p_1,2} & b_{p_1,3} & \dots & b_{p_1,10} & b_{p_1,11} & b_{p_1,12} \\ b_{p_2,1} & b_{p_2,2} & b_{p_1,3} & \dots & b_{p_2,10} & b_{p_2,11} & b_{p_2,12} \\ b_{p_3,1} & b_{p_3,2} & b_{p_3,3} & \dots & b_{p_3,10} & b_{p_3,11} & b_{p_3,12} \end{bmatrix}, \tag{27}$$

where

$$\begin{aligned} &\text{for} \quad I = \rho \qquad (p_1, p_2, p_3) \equiv (1, 5, 9) \\ &\text{for} \quad I = u \qquad (p_1, p_2, p_3) \equiv (2, 6, 10) \\ &\text{for} \quad I = v \qquad (p_1, p_2, p_3) \equiv (3, 7, 11) \\ &\text{for} \quad I = e \qquad (p_1, p_2, p_3) \equiv (4, 8, 12). \end{aligned} \tag{28}$$

The norms of these submatrices, used in Eq. (26), are computed by $\|\boldsymbol{b}_I\| = \max_{1 \le j \le n_{ee}} \{|b_{p_1,j}| + |b_{p_2,j}| + |b_{p_3,j}|\}$.

The solution is advanced in time by the implicit predictor-multicorrector algorithm given in [11]. The resulting linear systems of equations are solved by a nodal-block-diagonal preconditioned GMRES method. All solutions in this work are obtained using a fixed time-step size with $CFL = 1$.

4 Numerical results

In this section we describe the 2D test computations carried out for two steady-state problems. The tolerance of the preconditioned GMRES algorithm is 0.1, the dimension of the Krylov subspace is 5, and the number of multicorrections is 3. All computations are initialized with the inflow values. The symbol τ_g represents τ calculated based on the element matrices, and the symbol τ_{dof} represents $\boldsymbol{\tau}$ calculated based on the degree-of-freedom submatrices of the element matrices.

4.1 Oblique shock

The first problem is a Mach 2 uniform flow over a wedge, at an angle of $-10°$ with respect to a horizontal wall. The solution involves an oblique shock at an angle of 29.3° emanating from the leading edge of the wedge, as shown in Fig. 1.

The computational domain is a square with $0 \le x \le 1$ and $0 \le y \le 1$. Prescribing the following inflow data on the left and top boundaries results in a solution with the following outflow data:

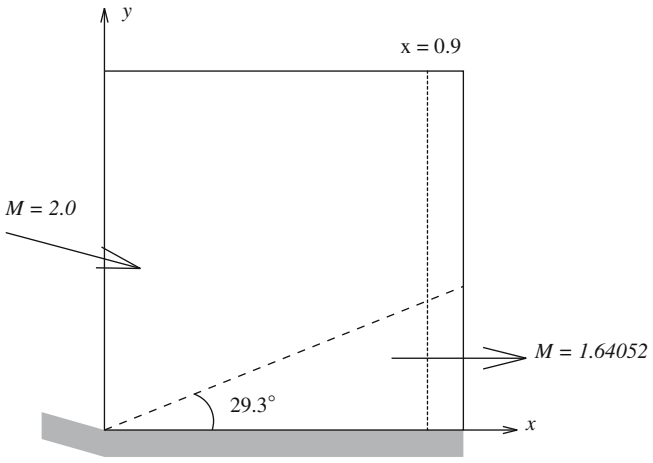
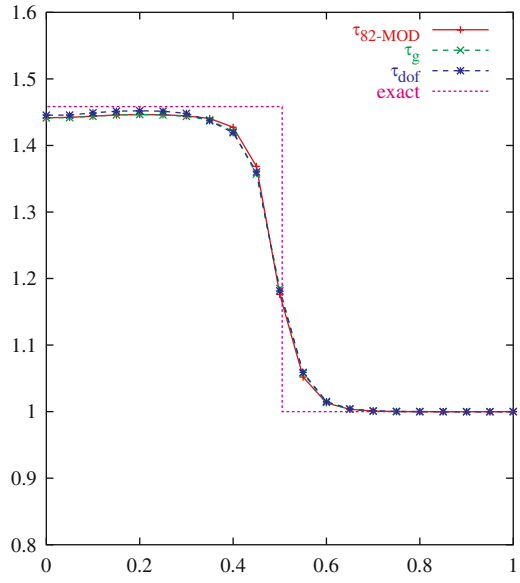


Fig. 1 Oblique shock – problem description

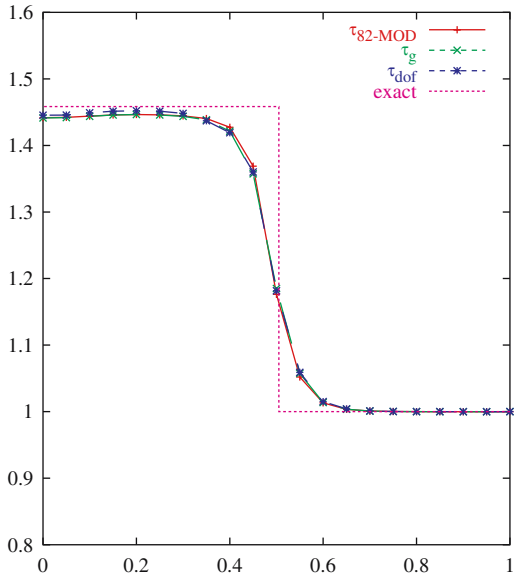
$$\text{Inflow} \begin{cases} M &= 2.0 \\ \rho &= 1.0 \\ u &= \cos\ 10^0 \\ v &= -\sin\ 10^0 \\ p &= 0.17857 \end{cases} \quad \text{Outflow} \begin{cases} M &= 1.64052 \\ \rho &= 1.45843 \\ u &= 0.88731\ . \\ v &= 0.0 \\ p &= 0.30475 \end{cases} \tag{29}$$

Here M is the Mach number and p is the pressure. Four Dirichlet boundary conditions are imposed at the left and top boundaries, the slip condition with $v = 0$ is set at the bottom boundary, and no boundary condition is imposed at the outflow (right) boundary. A 20×20 mesh with 800 linear triangles and 441 nodes is employed.

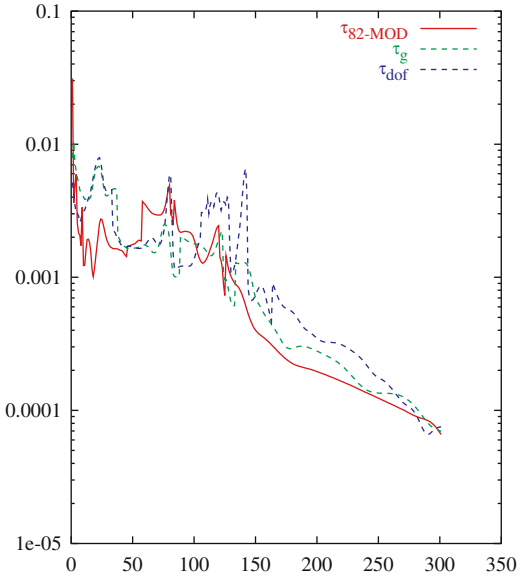
Figure 2a,c show, respectively, the density along line $x = 0.9$ and the evolution of the density residual for 300 steps, computed with $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, with $r = 2$. Here we update τ_g, τ_{dof} and δ_{91} at every nonlinear iteration of a time level (i.e. iteration update). We also tested the case where we update τ_g, τ_{dof} and δ_{91} only at the beginning of every time step (i.e. time-step update). The results are shown in Fig. 2b,d. The evolution of the density residual is faster for time-step update than it is for iteration update. Figure 3 shows the influence of an algorithmic feature [3], which is based on freezing the shock-capturing parameter δ_{91} at its current value when a convergence stagnation is detected. Table 1 shows the number of GMRES iterations (N_{GMRES}) and number of steps (N_{steps}) corresponding to Fig. 3. The computations with τ_{dof} need less steps than the other alternatives, but more GMRES iterations than it is needed with τ_g. However, the $\boldsymbol{\tau}_{\text{dof}}$ computations are less costly than the τ_g computations. That is, the calculations required by Eqs. (24), (25) and (26), which involve norms of element submatrices, are less costly than the calculations required by Eqs. (22) and (23), which involve norms of element matrices. We also note that $\tau_{82\text{-MOD}}$ needs more steps than all others. The relative performances with the time-step update are similar, but less steps are required.



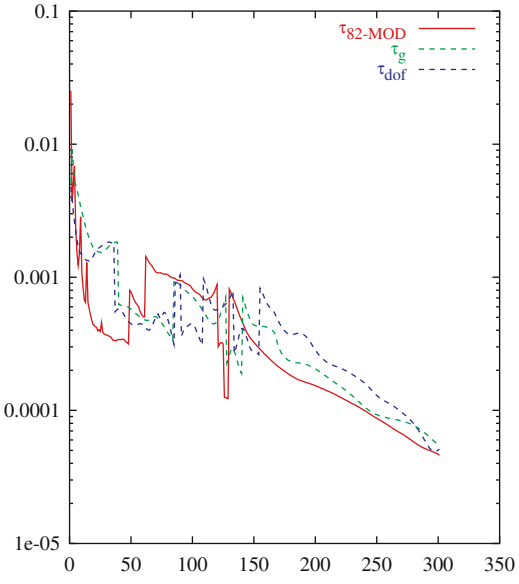
(a) Density profile at $x = 0.9$ – iteration update



(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

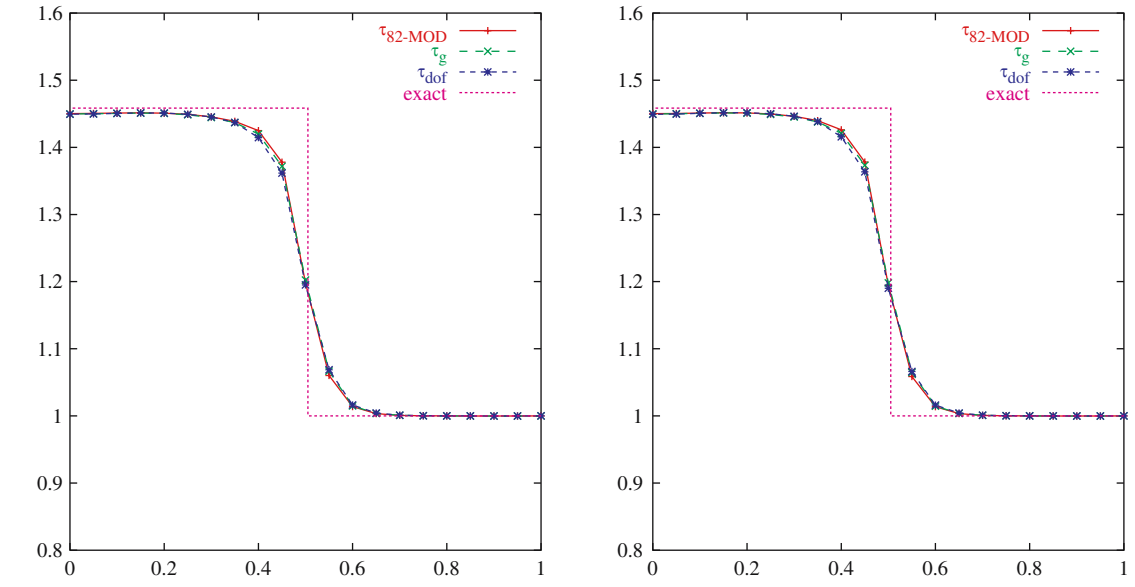
Fig. 2 Oblique shock – solutions and residuals obtained with $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, and the iteration update and time-step update. **a** Density profile at $x = 0.9$ – iteration update. **b** Density profile at $x = 0.9$ – time-step update. **c** Evolution of density residual – iteration update. **d** Evolution of density residual – time-step update

Table 1 Oblique shock–computational costs (in number of GMRES iterations and time steps) for $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, with iteration update and time-step update (both with freezing the shock-capturing parameter)

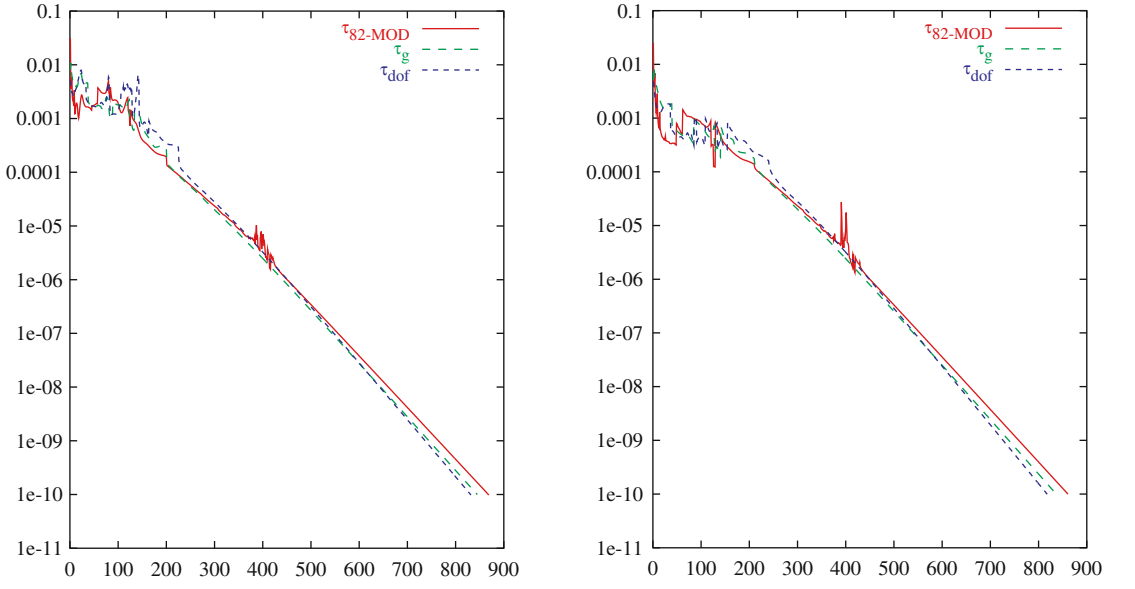
Update	$\tau_{82\text{-MOD}}$		τ_g		τ_{dof}	
	N_{GMRES}	N_{steps}	N_{GMRES}	N_{steps}	N_{GMRES}	N_{steps}
Iteration	5,226	869	4,287	846	4,902	833
Time-step	5,289	861	4,246	838	4,857	819

4.2 Reflected shock

This problem consists of three regions (R1, R2, R3) separated by an oblique shock and its reflection from a wall, as shown in Fig. 4. Prescribing the following Mach 2.9 inflow data in the first region on the left (R1), and requiring the incident shock to be at an angle of 29°, leads to the following exact solution at the other two regions (R2, R3):



(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

Fig. 3 Oblique shock – solutions and residuals obtained with $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, and the iteration update and time-step update (both with freezing the shock-capturing parameter). **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

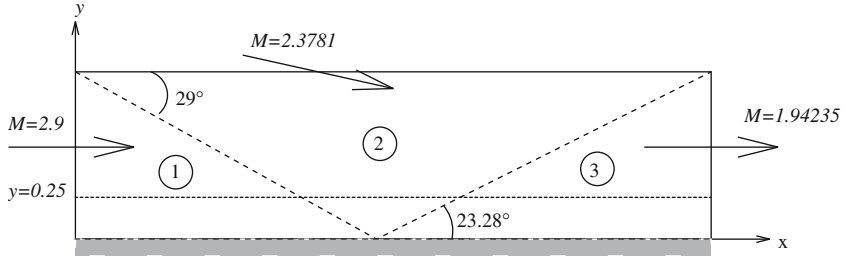
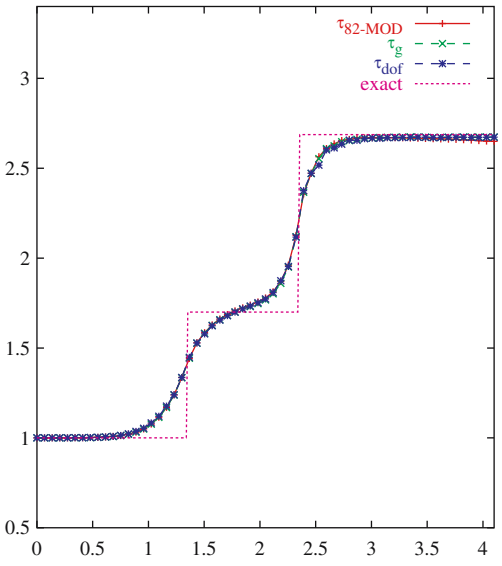
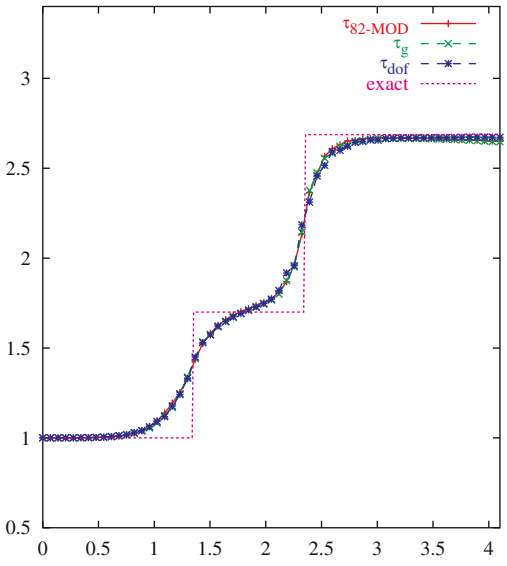


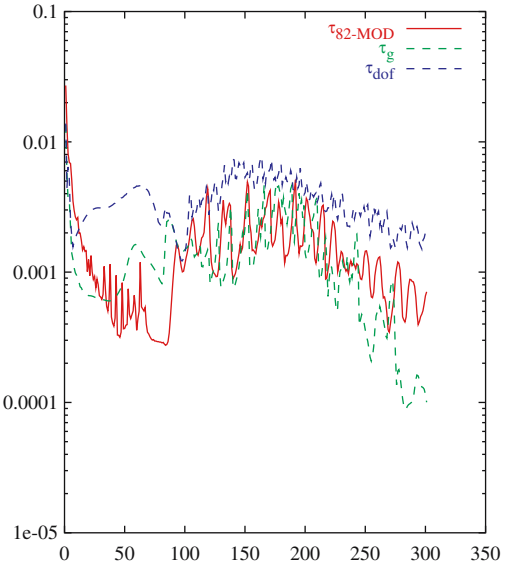
Fig. 4 Reflected shock – problem description



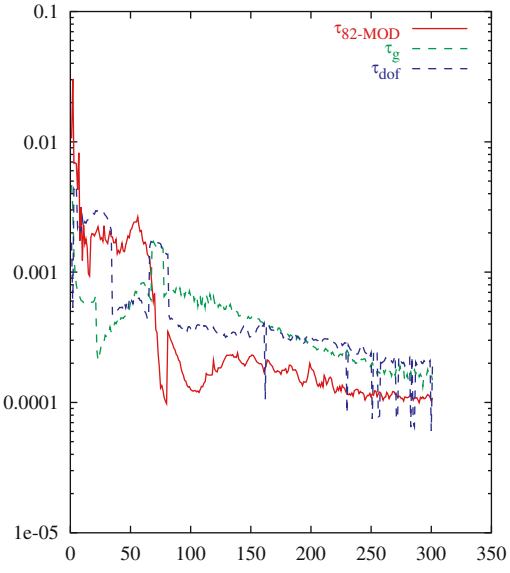
(a) Density profile at $y = 0.25$ – iteration update.



(b) Density profile at $y = 0.25$ – time-step update.



(c) Evolution of density residual – iteration update.



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

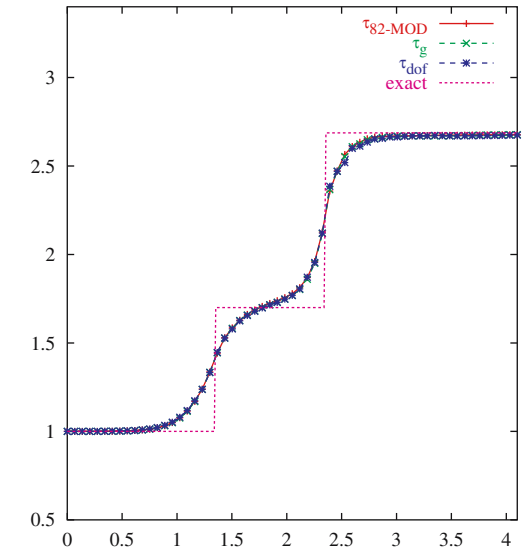
Fig. 5 Reflected shock (with uniform mesh) – solutions and residuals obtained with $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, and the iteration update and time-step update. **a** Density profile at $y = 0.25$ – iteration update. **b** Density profile at $y = 0.25$ – time-step update. **c** Evolution of density residual – iteration update **d** Evolution of density residual – time-step update.

$$
\text{R1}\begin{cases} M = 2.9 \\ \rho = 1.0 \\ u = 2.9 \\ v = 0.0 \\ p = 0.714286 \end{cases}
\quad
\text{R2}\begin{cases} M = 2.3781 \\ \rho = 1.7 \\ u = 2.61934 \\ v = -0.50632 \\ p = 1.52819 \end{cases}
$$

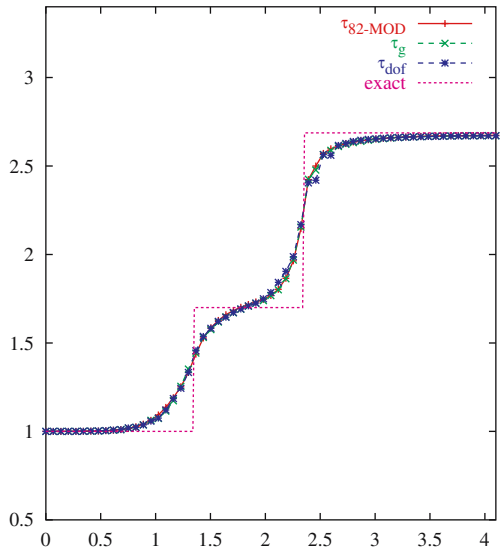
$$
\text{R3}\begin{cases} M = 1.94235 \\ \rho = 2.68728 \\ u = 2.40140 \\ v = 0.0 \\ p = 2.93407 \end{cases} \tag{30}
$$

The computational domain is a rectangle with $0 \leq x \leq 4.1$ and $0 \leq y \leq 1$. We prescribe the density, velocities and pressure at the left and top boundaries, the slip condition with $v = 0$ is imposed at the bottom boundary, and no boundary condition is imposed at the outflow (right) boundary. We use a uniform mesh with 60×20 cells, where each cell is divided into two triangles (1,281 nodes and 2,400 elements), and an unstructured mesh with 1,837 nodes and 3,429 elements.

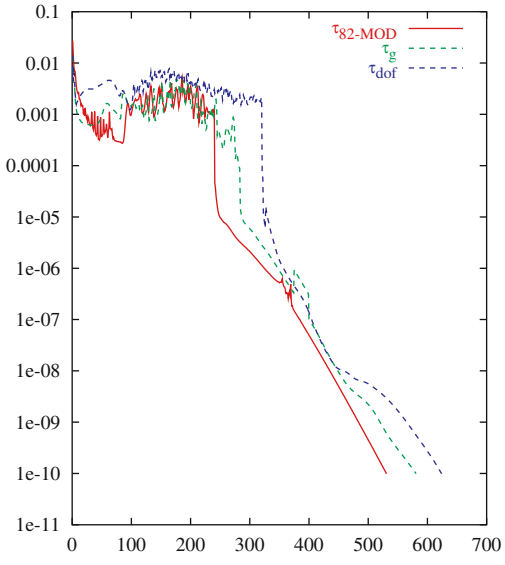
Figure 5a,b show the density along line $y = 0.25$ for the uniform mesh, computed with $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, with $r = 2$,



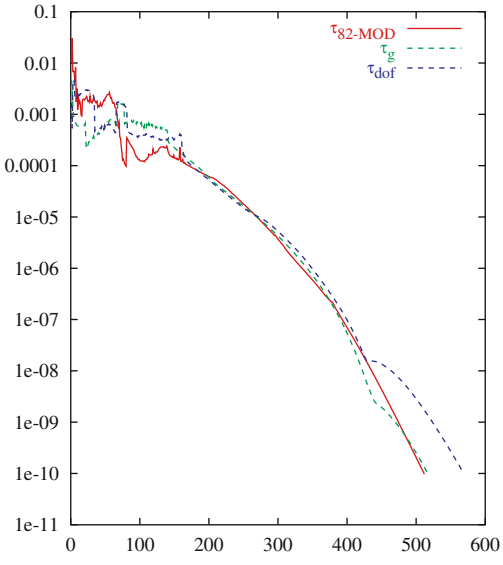
(a) Density profile at $y = 0.25$ – iteration update



(b) Density profile at $y = 0.25$ – time-step update



(c) Evolution of density residual – iteration update



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

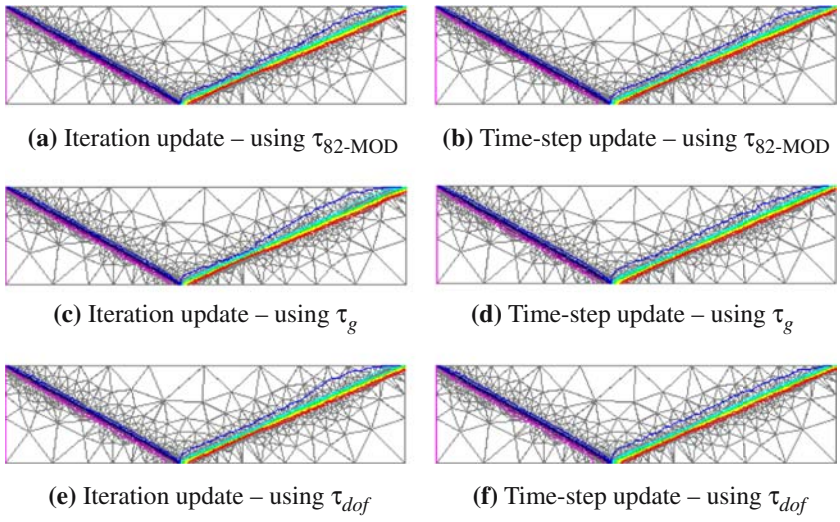
Fig. 6 Reflected shock (with uniform mesh) – solutions and residuals obtained with $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, and the iteration update and time-step update (both with freezing the shock-capturing parameter). **a** Density profile at $y = 0.25$ – iteration update. **b** Density profile at $y = 0.25$ – time-step update. **c** Evolution of density residual – iteration update. **d** Evolution of density residual – time-step update

for iteration update and time-step update. The density profile with τ_{dof} and time-step update shows some oscillations. The evolutions of the density residual for 300 steps are shown in Figure 5c,d. We note that the residuals are smaller with the time-step update. Figure 6 shows the influence of the algorithmic feature based on freezing the shock-capturing parameter. We note that in this case $\tau_{82\text{-MOD}}$ leads to somewhat smaller residuals. We also see in Table 2 that $\tau_{82\text{-MOD}}$ requires less steps for both iterations update and time-step update. The performances with τ_g and τ_{dof} are more sensitive to which update strategy is used, with better performance for the time-step update.

Table 2 Reflected shock (with uniform mesh) – computational costs (in number of GMRES iterations and time steps) for $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, with iteration update and time-step update (both with freezing the shock-capturing parameter).

Update	$\tau_{82\text{-MOD}}$		τ_g		τ_{dof}	
	N_{GMRES}	N_{steps}	N_{GMRES}	N_{steps}	N_{GMRES}	N_{steps}
Iteration	3,226	531	3,368	582	3,743	625
Time-step	3,180	512	3,073	518	3,382	570

Figures 7 and 8 show the results for the unstructured mesh, computed with the same set of τs and update strategies. The algorithmic feature based on freezing the shock-capturing

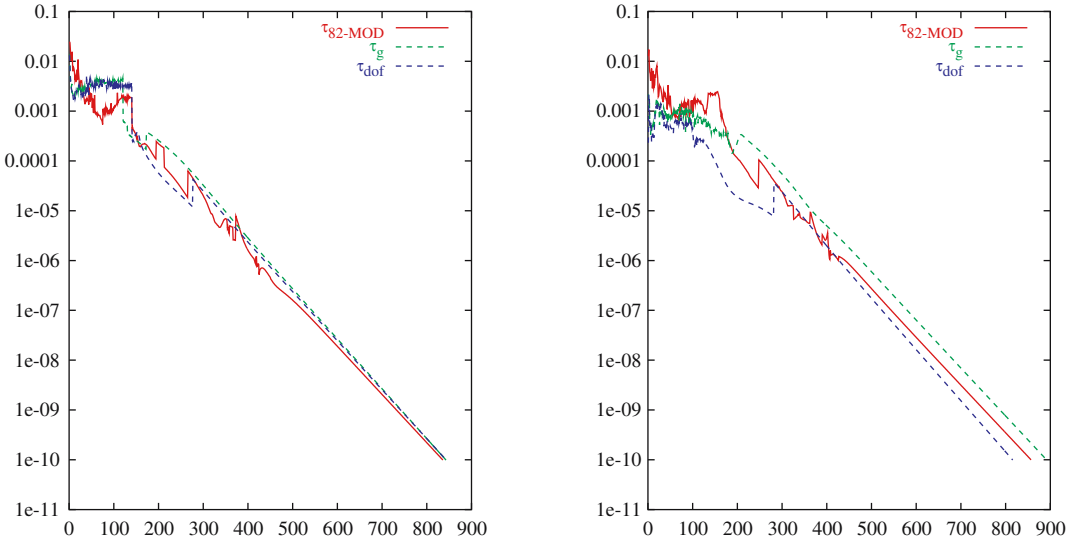


(a) Iteration update – using $\tau_{82\text{-MOD}}$ **(b)** Time-step update – using $\tau_{82\text{-MOD}}$

(c) Iteration update – using τ_g **(d)** Time-step update – using τ_g

(e) Iteration update – using τ_{dof} **(f)** Time-step update – using τ_{dof}

Fig. 7 Reflected shock – density computed with $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, and the iteration update and time-step update (both with freezing the shock-capturing parameter). **a** Iteration update – using $\tau_{82\text{-MOD}}$. **b** Time-step update – using $\tau_{82\text{-MOD}}$. **c** Iteration update – using τ_g. **d** Time-step update – using τ_g. **e** Iteration update – using τ_{dof}. **f** Time-step update – using τ_{dof}



(a) Evolution of density residual – iteration update. **(b)** Evolution of density residual – time-step update.

Fig. 8 Reflected shock (with unstructured mesh) – residuals obtained with $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, and the iteration update and time-step update (both with freezing the shock-capturing parameter). **a** Evolution of density residual – iteration update. **b** Evolution of density residual – time-step update.

Table 3 Reflected shock (with unstructured mesh) – computational costs (in number of GMRES iterations and time steps) for $\tau_{82\text{-MOD}}$, τ_g and τ_{dof}, with iteration update and time-step update (both with freezing the shock-capturing parameter).

Update	$\tau_{82\text{-MOD}}$		τ_g		τ_{dof}	
	N_{GMRES}	N_{steps}	N_{GMRES}	N_{steps}	N_{GMRES}	N_{steps}
Iteration	9,573	836	9,127	844	9,138	845
Time-step	9,977	857	9,913	893	9,278	817

parameter is activated. The density distributions are in good agreement. Looking at the number of GMRES iterations in Table 3, we see that for iteration update τ_g and τ_{dof} are quite comparable and both better than $\tau_{82\text{-MOD}}$. For time-step update τ_{dof} is the best.

5 Concluding remarks

For the SUPG formulation of inviscid compressible flows, we described stabilization parameters that are defined for each degree-of-freedom of each element. These definitions are expressed based on the norms of the degree-of-freedom submatrices of the element-level matrices. They take into account the flow field, the local length scales, and the time

step size. We carried out a number of 2D test computations for steady-state problems involving supersonic flows and shocks. By inspecting the solution quality and convergence history, we investigated the performance differences between the τ definitions described here, the τ definitions based on the full element-level matrices, and $\tau_{82\text{-MOD}}$. In all cases the formulation includes a shock-capturing term involving the shock-capturing parameter δ_{91}. Also by inspecting the solution quality and convergence history, we investigated the performance difference between updating the stabilization and shock-capturing parameters at time level n and at (every nonlinear iteration of) time level $n + 1$. The formulation includes, as an option, 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 degree-of-freedom submatrices and full element-level matrices show comparable performances.

References

1. Brooks AN, Hughes TJR (1982) Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations. Comput Methods Appl Mech Eng 32:199–259
2. Catabriga L, Coutinho ALGA (2002a) Implicit SUPG solution of Euler equations using edge-based data structures. Comput Methods Appl Mech Eng 191:3477–3490
3. Catabriga L, Coutinho ALGA (2002b) Improving convergence to steady-state of implicit-SUPG solution of Euler equations. Commun Numer Methods Eng 18:345–353
4. Catabriga L, Coutinho ALGA, Tezduyar TE (2002) Finite element SUPG parameters computed from local matrices for compressible flows. In: Proceedings of the 9th Brazilian congress of engineering and thermal sciences. Caxambu, Brazil
5. Catabriga L, Coutinho ALGA, Tezduyar TE (2003a) Finite element SUPG parameters computed from local dof-matrices for compressible flows. In: Proceedings of the 24th Iberian Latin-American congress on computational methods in engineering. Ouro Preto, Brazil
6. Catabriga L, Coutinho ALGA, Tezduyar TE (2003b) Finite element SUPG parameters computed from local edge matrices for compressible flows. In: Proceedings of the 17th international congress of mechanical engineering. Sao Paulo, Brazil
7. Catabriga L, Coutinho ALGA, Tezduyar TE (2004) Compressible flow SUPG stabilization parameters computed from element-edge matrices. Comput Fluid Dyn J 13:450–459
8. Catabriga L, Coutinho ALGA, Tezduyar TE (2005) Compressible flow SUPG parameters computed from element matrices. Commun Numer Methods Eng 21:465–476
9. Donea J (1984) A Taylor–Galerkin method for convective transport problems. Int J Numer Methods Eng 20:101–120
10. Hughes TJR, Brooks AN (1979) A multi-dimensional upwind scheme with no crosswind diffusion. In: Hughes TJR (ed) finite element methods for convection dominated flows, AMD-vol 34. ASME, New York, pp 19–35
11. Hughes TJR, Tezduyar TE (1984) Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations. Comput Methods Appl Mech Eng 45:217–284
12. Hughes TJR, Franca LP, 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. Comput Methods Appl Mech Eng 63:97–112
13. Johnson C, Navert U, Pitkäranta J (1984) Finite element methods for linear hyperbolic problems. Comput Methods Appl Mech Eng 45:285–312
14. Le Beau GJ, Tezduyar TE (1991) Finite element computation of compressible flows with the SUPG formulation. In: Advances in finite element analysis in fluid dynamics, FED-vol. 123. ASME, New York, pp 21–27
15. Tezduyar TE (1992) Stabilized finite element formulations for incompressible flow computations. Adv Appl Mech 28:1–44
16. Tezduyar TE (2001) Adaptive determination of the finite element stabilization parameters. In: Proceedings of the ECCOMAS computational fluid dynamics conference 2001 (CD-ROM). Swansea, Wales
17. Tezduyar TE (2002) Calculation of the stabilization parameters in SUPG and PSPG formulations. In: Proceedings of the 1st South-American congress on computational mechanics (CD-ROM). Santa Fe–Parana, Argentina
18. Tezduyar TE (2003) Computation of moving boundaries and interfaces and stabilization parameters. Int J Numer Methods Fluids 43:555–575
19. Tezduyar TE, Hughes TJR (1982) Development of time-accurate finite element techniques for first-order hyperbolic systems with particular emphasis on the compressible Euler equations. NASA Technical Report NASA-CR-204772, NASA. http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19970023187_19970349 54.pdf
20. Tezduyar TE, Hughes TJR (1983) Finite element formulations for convection dominated flows with particular emphasis on the compressible Euler equations. In: Proceedings of AIAA 21st aerospace sciences meeting. AIAA Paper 83-0125, Reno, Nevada
21. Tezduyar TE, Osawa Y (2000) Finite element stabilization parameters computed from element matrices and vectors. Comput Methods Appl Mech Eng 190:411–430
22. Tezduyar TE, Park YJ (1986) Discontinuity capturing finite element formulations for nonlinear convection-diffusion-reaction equations. Comput Methods Appl Mech Eng 59:307–325
23. Tezduyar T, Aliabadi S, Behr M, Johnson A, Mittal S (1993) Parallel finite-element computation of 3D flows. Computer 26(10): 27–36