A Nonlinear Multiscale Viscosity Method to Solve Compressible Flow Problems

Sérgio Souza Bento(✉), Leonardo Muniz de Lima, Ramoni Zancanela Sedano,
Lucia Catabriga, and Isaac P. Santos

High Performance Computing Lab, Federal University of Espírito Santo,
Av. Fernando Ferrari, 514, Goiabeiras, Vitória, ES 29075-910, Brazil
{sergio.bento,isaac.santos}@ufes.br, lmuniz@ifes.edu.br,
{rsedano,luciac}@inf.ufes.br

Abstract. In this work we present a nonlinear multiscale viscosity method to solve inviscid compressible flow problems in conservative variables. The basic idea of the method consists of adding artificial viscosity adaptively in all scales of the discretization. The amount of viscosity added to the numerical model is based on the YZβ shock-capturing parameter, which has the property of being mesh and numerical solution dependent. The subgrid scale space is defined using bubble functions whose degrees of freedom are locally eliminated in favor of the degrees of freedom that live on the resolved scales. This new numerical formulation can be considered a free parameter and self adaptive method. Performance and accuracy comparisons with the well known method SUPG combined with shock capturing operators are conducted based on benchmark 2D problems.

Keywords: Finite element method · Multiscale stabilized formulation · Compressible flow problems

1 Introduction

The numerical solution of the compressible flows may exhibit global spurious oscillations, especially near shock regions. More accurate and stable results can be obtained using stabilized formulations, either linear or nonlinear approach [4,5,12,20]. In the 1990s it was shown that the stabilized finite element methods could be derived from the variational multiscale framework, which consists of a consistent decomposition of the approximation space into resolved (coarse) and unresolved (subgrid) scales subspaces via a variational projection. The numerical oscillations originated by the standard Galerkin method can be related to scales that are not represented by the discretization, that is, the unresolved scales. In this case, those unresolved scales or their effect may be inserted into the problem formulation to be solved on the resolved scales, represented by the chosen discretization. Examples of multiscale methods can be found in [6,9,11,13,14]. It is important to highlight that those stabilization/multiscale techniques prevent numerical oscillations and other instabilities in solving problems with high Reynolds and/or Mach numbers and shocks or strong boundary layers [17].

O. Gervasi et al. (Eds.): ICCSA 2016, Part I, LNCS 9786, pp. 3–17, 2016.
DOI: 10.1007/978-3-319-42085-1_1

Santos and Almeida [15] presented a nonliner multiscale method to solve advection dominated transport problem where a nonlinear artificial diffusion is added on the subgrid scale. The amount of artificial diffusion added to the numerical model was stablished considering a two-level decomposition of the function space and the velocity field into the resolved (coarse) and unresolved (subgrid) scales. The subgrid velocity field was determined by requiring the minimum of the associated kinetic energy for which the residue of the resolved scale solution vanishes on each element of the discretization. The idea of adding a nonlinear diffusion in both scales (subgrid and coarse) of the discretization was considered in [2] through the Dynamic Diffusion (DD) method, where the subgrid space is constructed by bubbles functions defined into elements and the amount of nonlinear diffusion is similar to the method presented in [15]. This methodology was extended to the compressible Euler equations in [16], where comparisons with the well known SUPG method coupled with the two shock capturing operators: the Consistent Approximate Upwind Petrov-Galerkin (CAU) [7] and the YZβ [19], were made. Although the DD method offered good results, it did not live up to expectations compared to the SUPG formulations with shock capturing operators as CAU and YZβ.

The SUPG method coupled with the YZβ shock capturing operator has offered numerical solution with good accuracy for compressible problems. Moreover, Tezduyar [17] has been proposing adaptive ways for calculations of the local length scale (also known as "element length") present in the stabilization parameters. The calculus of the local length scale parameter is made taking into account the directions of high gradients and the spatial discretization domain. The stabilization parameter resulting acts adaptively and is useful to avoid excessive viscosity helping to maintain smaller numerical dissipations.

In this paper we propose a new numerical formulation, named, Nonlinear Multiscale Viscosity (NMV) method, to solve inviscid compressible flow problems in conservative variables. As the DD method, the basic idea is to add a nonlinear artificial viscosity in all scales of the discretization, but the amount of artificial viscosity is defined by the stabilization parameter of the YZβ method, as proposed in [19,20]. The nonlinear artificial viscosity added to the numerical formulation is made adaptively, leading the NMV to a self adaptive methodology.

The remainder of this work is organized as follows. Section 2 briefly addresses the governing equations and the variational multiscale formulation. Numerical experiments are conducted in Sect. 3 to show the behavior of the new multiscale finite element method for a variety of benchmark Euler equations problems. Section 4 concludes this paper.

2 Governing Equations and Variational Multiscale Formulation

The two-dimensional Euler equations in conservative variables, $\mathbf{U} = (\rho, \rho u, \rho v, \rho e)$, without source terms are an inviscid system of conservation laws represented by

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

where 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 $\mathrm{I\!R}^2$, and T_f is a positive real number, representing the final time. Alternatively, Eq. (1) can be written as

$$\frac{\partial \mathbf{U}}{\partial t} + \mathbf{A_x}\frac{\partial \mathbf{U}}{\partial x} + \mathbf{A_y}\frac{\partial \mathbf{U}}{\partial y} = \mathbf{0}, \quad \text{on} \quad \Omega \times [0, T_f], \tag{2}$$

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

To define the finite element discretization, we consider a triangular partition $\mathcal{T}_H$ of the domain Ω into n_{el} elements, where: $\Omega = \bigcup_{e=1}^{n_{el}} \Omega_e$ and $\Omega_i \cap \Omega_j = \emptyset$, $i, j = 1, 2, \cdots, n_{el}$, $i \neq j$. We introduce the space $\mathcal{V}_E$, that is written as the direct sum

$$\mathcal{V}_E = \mathcal{V}_h \oplus \mathcal{V}_B, \tag{3}$$

where the subspaces $\mathcal{V}_h$ and $\mathcal{V}_B$ are given by

$$\mathcal{V}_h = \{\mathbf{U}_h \in [H^1(\Omega)]^4 \quad | \quad \mathbf{U}_h|_{\Omega_e} \in [\mathbb{P}_1(\Omega_e)]^4, \mathbf{U}_h \cdot \mathbf{e}_k = g_k(t) \;\; \text{in} \;\; \Gamma_{g_k}\};$$

$$\mathcal{V}_B = \{\mathbf{U}_B \in [H^1_0(\mathcal{T}_H)]^4 \quad | \quad \mathbf{U}_B|_{\Omega_e} \in [span(\psi_B)]^4, \quad \forall \Omega_e \in \mathcal{T}_H\},$$

where $\mathbb{P}_1(\Omega_e)$ represents the set of first order polynomials in Ω_e, ψ_B is a bubble function ($0 \leq \psi_B \leq 1$ and $\psi_B \in H^1_0(\mathcal{T}_H)$) and H^1, H^1_0 are Hilbert spaces [3]. The space $\mathcal{V}_h$ represents the resolved (coarse) scale space whereas $\mathcal{V}_B$ stands for the subgrid (fine) scale space (Fig. 1).

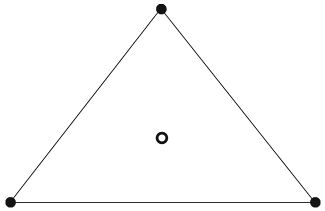


Fig. 1. $\mathcal{V}_E$ Representation: $\bullet$ stands for $\mathcal{V}_h$ nodes and $\circ$ stands for $\mathcal{V}_B$ nodes.

The NMV method for the Euler equation consists of find $\mathbf{U}_E = \mathbf{U}_h + \mathbf{U}_B \in \mathcal{V}_E$ with $\mathbf{U}_h \in \mathcal{V}_h$, $\mathbf{U}_B \in \mathcal{V}_B$ such that

$$\int_\Omega \mathbf{W}_E \cdot \Big(\frac{\partial \mathbf{U}_E}{\partial t} + \mathbf{A}^h_x \frac{\partial \mathbf{U}_E}{\partial x} + \mathbf{A}^h_y \frac{\partial \mathbf{U}_E}{\partial y}\Big) d\Omega +$$

$$\sum_{e=1}^{nel} \int_{\Omega_e} \delta_h \Big(\frac{\partial \mathbf{W}_E}{\partial x} \cdot \frac{\partial \mathbf{U}_E}{\partial x} + \frac{\partial \mathbf{W}_E}{\partial y} \cdot \frac{\partial \mathbf{U}_E}{\partial y}\Big) d\Omega = \mathbf{0} \;\; \forall \mathbf{W}_E \in \mathcal{V}_E, \tag{4}$$

where $\mathbf{W}_E = \mathbf{W}_h + \mathbf{W}_B \in \mathcal{V}_E$ with $\mathbf{W}_h \in \mathcal{V}_h$, $\mathbf{W}_B \in \mathcal{V}_B$ and the amount of artificial viscosity is calculated on the element-level by using the YZβ shock-capturing viscosity parameter [19]

$$\delta_h = \|\mathbf{Y}^{-1}R(\mathbf{U}_h)\| \left(\sum_{i=1}^{2} \left\| \mathbf{Y}^{-1} \frac{\partial \mathbf{U}_h}{\partial x_i} \right\|^2 \right)^{\frac{\beta}{2}-1} \|\mathbf{Y}^{-1}\mathbf{U}_h\|^{1-\beta} h^{\beta}, \tag{5}$$

where

$$R(\mathbf{U}_h) = \frac{\partial \mathbf{U}_h}{\partial t} + \mathbf{A}_x^h \frac{\partial \mathbf{U}_h}{\partial x} + \mathbf{A}_y^h \frac{\partial \mathbf{U}_h}{\partial y}$$

is the residue of the problem on Ω_e, $\mathbf{Y}$ is a diagonal matrix constructed from the reference values of the components of $\mathbf{U}$, given by

$$\mathbf{Y} = diag\left((U_1)_{\text{ref}}, (U_2)_{\text{ref}}, (U_3)_{\text{ref}}, (U_4)_{\text{ref}}\right), \tag{6}$$

h is the local length scale defined as in [17] by

$$h = \left(\sum_a |\mathbf{j} \cdot \nabla N_a| \right)^{-1}, \tag{7}$$

$\mathbf{j}$ is a unit vector defined as

$$\mathbf{j} = \frac{\nabla \rho}{\|\nabla \rho\|}$$

and N_a is the interpolation function associated with node a. It is important to note that, the local length h is defined automatically taking into account the directions of high gradients and spatial discretization domain.

Generally, the parameter β is set as $\beta = 1$ for smoother shocks and $\beta = 2$ for sharper shocks. The compromise between the $\beta = 1$ and $\beta = 2$ selections was defined in [17–19] as the following average expression for δ_h:

$$\delta_h = \frac{1}{2} \left(\delta_h|_{\beta=1} + \delta_h|_{\beta=2} \right).$$

The numerical solution is obtained using iterative procedures for space and time. The iterative procedure for space is defined of the following way: given $\mathbf{U}_E^i$ at iteration i, we find $\mathbf{U}_E^{i+1}$ satisfying the formulation (4) with $\delta_h = \delta_h(\mathbf{U}_E^i) = \delta_h^i$, for $i = 0, 1, \cdots, i_{MAX}$. The formulation (4) can be partitioned in two subproblems, one related to the resolved scale, given by

$$\int_{\Omega} \mathbf{W}_h \cdot \Big(\frac{\partial \mathbf{U}_h^{i+1}}{\partial t} + \mathbf{A}_x^h \frac{\partial \mathbf{U}_h^{i+1}}{\partial x} + \mathbf{A}_y^h \frac{\partial \mathbf{U}_h^{i+1}}{\partial y} \Big) d\Omega +$$

$$\int_{\Omega} \mathbf{W}_h \cdot \Big(\frac{\partial \mathbf{U}_B^{i+1}}{\partial t} + \mathbf{A}_x^h \frac{\partial \mathbf{U}_B^{i+1}}{\partial x} + \mathbf{A}_y^h \frac{\partial \mathbf{U}_B^{i+1}}{\partial y} \Big) d\Omega +$$

$$\sum_{e=1}^{nel} \int_{\Omega_e} \delta_h^i \Big(\frac{\partial \mathbf{W}_h}{\partial x} \cdot \frac{\partial \mathbf{U}_h^{i+1}}{\partial x} + \frac{\partial \mathbf{W}_h}{\partial y} \cdot \frac{\partial \mathbf{U}_h^{i+1}}{\partial y} \Big) d\Omega = \mathbf{0}, \quad \forall \mathbf{W}_h \in \mathcal{V}_h, \tag{8}$$

and another, representing the subgrid scale is written as

$$\int_{\Omega} \mathbf{W}_B \cdot \frac{\partial \mathbf{U}_B^{i+1}}{\partial t} d\Omega + \int_{\Omega} \mathbf{W}_B \cdot \Big(\frac{\partial \mathbf{U}_h^{i+1}}{\partial t} + \mathbf{A}_x^h \frac{\partial \mathbf{U}_h^{i+1}}{\partial x} + \mathbf{A}_y^h \frac{\partial \mathbf{U}_h^{i+1}}{\partial y} \Big) d\Omega +$$

$$\sum_{e=1}^{nel} \int_{\Omega_e} \delta_h^i \Big(\frac{\partial \mathbf{W}_B}{\partial x} \cdot \frac{\partial \mathbf{U}_B^{i+1}}{\partial x} + \frac{\partial \mathbf{W}_B}{\partial y} \cdot \frac{\partial \mathbf{U}_B^{i+1}}{\partial y} \Big) d\Omega = \mathbf{0}, \quad \forall \mathbf{W}_B \in \mathcal{V}_B, \qquad (9)$$

where some terms were omitted, once they are zero.

Applying the standard finite element approximation on Eqs. (8) and (9), we arrive at a local system of ordinary differential equations:

$$\begin{bmatrix} M_{hh} & M_{hB} \\ M_{Bh} & M_{BB} \end{bmatrix} \begin{bmatrix} \dot{U}_h \\ \dot{U}_B \end{bmatrix} + \begin{bmatrix} K_{hh} & K_{hB} \\ K_{Bh} & K_{BB} \end{bmatrix} \begin{bmatrix} U_h \\ U_B \end{bmatrix} = \begin{bmatrix} 0_h \\ 0_B \end{bmatrix}, \qquad (10)$$

where U_h and U_B are, respectively, the nodal values of the unknowns $\mathbf{U}_h$ and $\mathbf{U}_B$ on each element Ω_e, whereas $\dot{U}_h$ and $\dot{U}_B$ are its time derivative.

The numerical solution is advanced in time by the implicit predictor-multicorrector algorithm given in [10] and adapted for the DD method in [16] for the Euler equations. The degrees of freedom related to the subgrid space are locally eliminated in favor of the ones of the macro space using a static condensation approach. Algorithm 1 shows the implicit predictor-multicorrector steps, considering second order approximations in time for the micro and macro scales subproblems, where Δt is the time-step; subscripts $n+1$ and n mean, respectively, the solution on the time-step $n+1$ and n; $\alpha = 0.5$ is the time advancing parameter; i is the iteration counter and N_2 is a nonsingular diagonal matrix. The resulting linear systems of equations are solved by the GMRES method considering all matrices stored by the well know strategy element-by-element [10].

3 Numerical Experiments

In this section we present numerical experiments considering three well known 2D benchmark problems: ‘oblique shock’, ‘reflected shock’ and ‘explosion’, discretized by unstructured triangular meshes using Delaunay triangulation through the software Gmsh [8]. The first and second problems used GMRES with 5 vectors to restart, tolerance equal to 10^{-1}, the number of multicorrections fixed to 3, the time-step size is 10^{-3} and the simulation is run until 3000 steps. The third problem used GMRES with 30 vectors to restart, tolerance equal to 10^{-5}, the number of multicorrections fixed to 3, the time-step size is 10^{-3} and the simulation is run until 250 steps. We compare the NMV method with SUPG formulation and two shock capturing operators, the CAU and the YZβ, named here, respectively, as SUPG + CAU and SUPG + YZβ. The tests were performed on a machine with an Intel Core i7-4770 3.4 GHz processor with 16 GB of RAM and Ubuntu 12.04 operating system.

Algorithm 1. NMV Predictor Multicorrector Algorithm

```
Step 1. $i = 0$

Step 2. Predictor phase:

$U_h^{n+1,0} = U_h^n + (1-\alpha)\Delta t\dot{U}_h^n,$
$\dot{U}_h^{n+1,0} = 0$
$U_B^{n+1,0} = U_B^n + (1-\alpha)\Delta t\dot{U}_B^n,$
$\dot{U}_B^{n+1,0} = 0$

Step 3. Multicorrector phase:

    Residual Force:
    $R_1^{n+1,i} = F_h^{n+1} - \left(M_{hh}\dot{U}_h^{n+1,i} + M_{hB}\dot{U}_B^{n+1,i}\right)$
    $\qquad - \left(K_{hh}U_h^{n+1,i} + K_{hB}U_B^{n+1,i}\right)$
    $R_2^{n+1,i} = F_B^{n+1} - \left(M_{Bh}\dot{U}_h^{n+1,i} + M_{BB}\dot{U}_B^{n+1,i}\right)$
    $\qquad - \left(K_{Bh}U_h^{n+1,i} + K_{BB}U_B^{n+1,i}\right)$

    Solve:
    $M^*\Delta\dot{U}_h^{n+1,i+1} = F^*,$
    with $M^* = M_1 - N_1N_2^{-1}M_2$ and $F^* = R_1 - N_1N_2^{-1}R_2$
    where $M_1 = M_{hh} + \alpha\Delta tK_{hh}$, $N_1 = M_{hB} + \alpha\Delta tK_{hB}$
    $M_2 = M_{Bh} + \alpha\Delta tK_{Bh}$ and $N_2 = M_{BB} + \alpha\Delta tK_{BB}$

    Corrector:
    $U_h^{n+1,i+1} = U_h^{n,i} + \alpha\Delta t\Delta\dot{U}_h^{n+1,i+1},$
    $\dot{U}_h^{n+1,i+1} = \dot{U}_h^{n+1,i} + \Delta\dot{U}_h^{n+1,i+1}$
    $U_B^{n+1,i+1} = U_B^{n,i} + \alpha\Delta t\Delta\dot{U}_B^{n+1,i+1},$
    $\dot{U}_B^{n+1,i+1} = \dot{U}_B^{n+1,i} + \Delta\dot{U}_B^{n+1,i+1}$

    with $\Delta\dot{U}_B^{n+1,i+1} = N_2^{-1}\left(R_2^{n+1,i} - M_2\Delta\dot{U}_h^{n+1,i+1}\right)$
```

3.1 2D Oblique Shock Problem

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. 2. 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:

$$\text{Inflow} \begin{cases} M &= 2.0 \\ \rho &= 1.0 \\ u &= \cos 10^0 \\ v &= -\sin 10^0 \\ p &= 0.17857 \end{cases} \qquad \text{Outflow} \begin{cases} M &= 1.64052 \\ \rho &= 1.45843 \\ u &= 0.88731 \\ v &= 0.0 \\ p &= 0.30475 \end{cases} \tag{11}$$

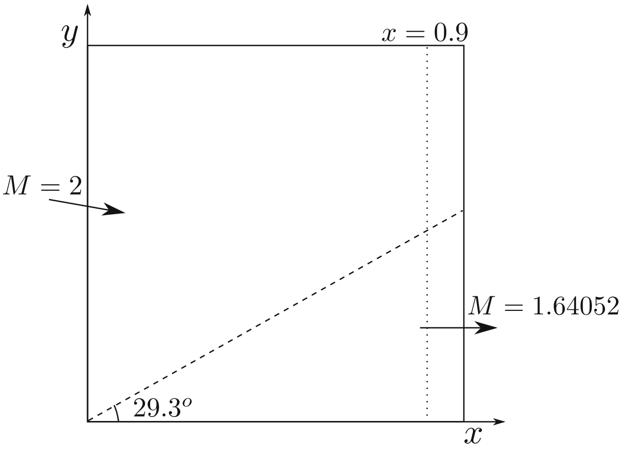


Fig. 2. Oblique shock problem description

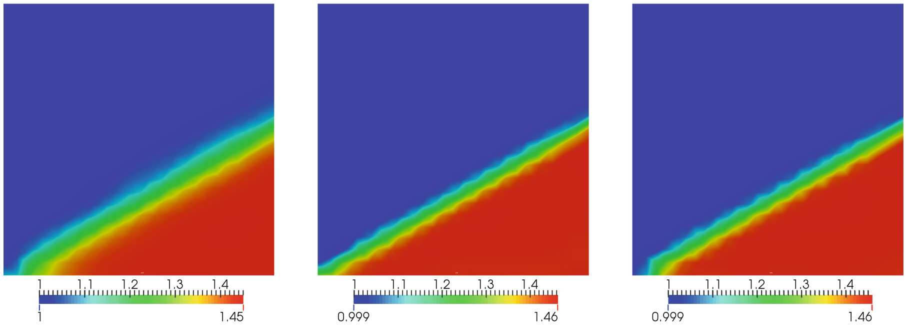
Here M is the Mach number and p is the pressure. Four Dirichlet boundary conditions are imposed at the left and the top boundaries, the condition $v = 0$ is set at the bottom boundary, and no boundary condition is imposed at the outflow (right) boundary.

For all simulations we consider an unstructured mesh consisting of 462 nodes and 846 elements. For the reference values used in Eq. (6), we consider the initial condition values for the left domain. Figure 3 shows the 2D density distribution obtained with all methods. Figure 4 shows the density profile along $x = 0.9$, obtained with SUPG + CAU, SUPG + YZβ and NMV methods. The solution obtained with the SUPG + YZβ is slightly better than the NMV on the left of the shock, whereas the solution with NMV is better on the right of the shock. The SUPG + CAU method clearly exhibit more dissipation.

On the other hand, the NMV method needs less GMRES iterations and CPU time than the others, as we can see in Table 1. The NMV method need less than half the number of GMRES iterations required by SUPG + CAU and SUPG + YZβ methods. Furthermore, the NMV method requires approximately 60 % and 55 %, respectively, of the CPU time required by the SUPG + CAU and the SUPG + YZβ methods.

Table 1. Oblique shock problem: computational performance

Methods	GMRES iterations	CPU time (s)
SUPG + CAU	58,514	54.225
SUPG + YZβ	69,863	60.302
NMV	18,020	33.361



(a) SUPG + CAU (b) SUPG + YZβ (c) NMV

Fig. 3. Oblique shock problem: density distribution 2D solution at time $t = 3$. (Color figure online)

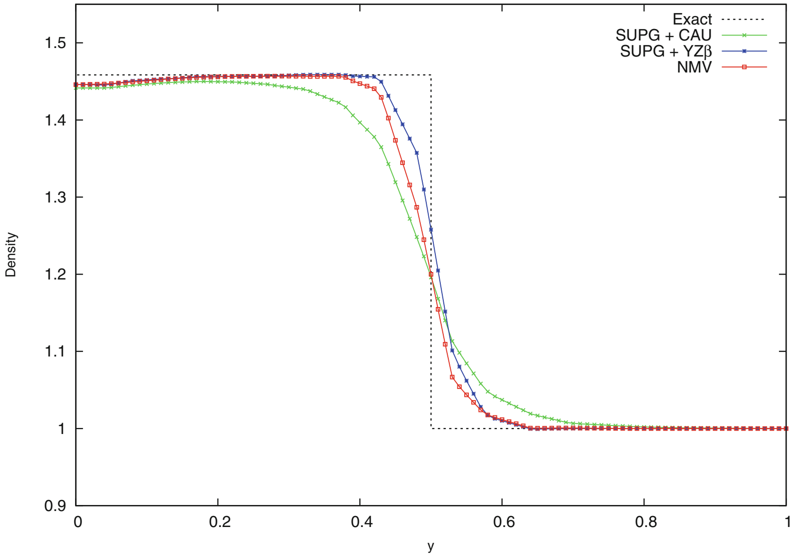


Fig. 4. Oblique shock problem: density profile along $x = 0.9$. (Color figure online)

3.2 2D Reflected Shock Problem

This 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 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 and R3):

$$\text{R1}\begin{cases} M &= 2.9 \\ \rho &= 1.0 \\ u &= 2.9 \\ v &= 0.0 \\ p &= 0.714286 \end{cases} \quad \text{R2}\begin{cases} M &= 2.3781 \\ \rho &= 1.7 \\ u &= 2.61934 \\ v &= -0.50632 \\ p &= 1.52819 \end{cases} \quad \text{R3}\begin{cases} M &= 1.94235 \\ \rho &= 2.68728 \\ u &= 2.40140 \\ v &= 0.0 \\ p &= 2.93407 \end{cases} \tag{12}$$

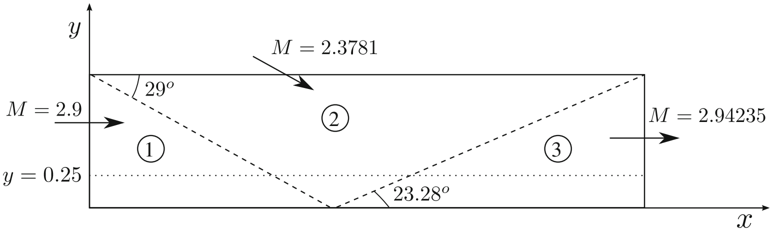


Fig. 5. Reflected shock problem description.

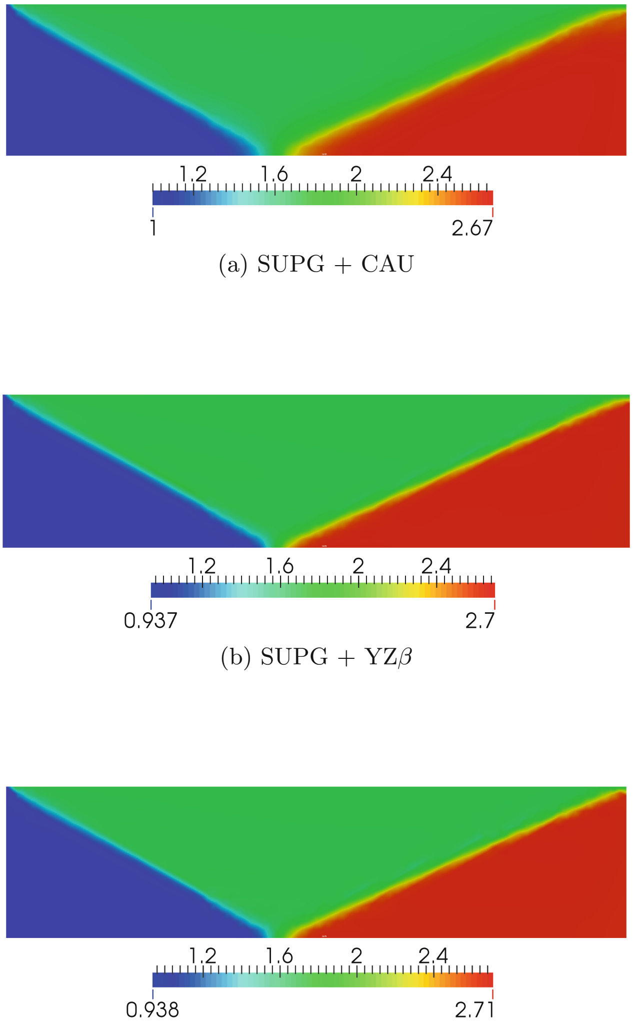
The computational domain is a rectangle with $0 \leq x \leq 4.1$ and $0 \leq y \leq 1$. We prescribe the density, velocities and pressure at the left and top boundaries, the slip condition with $v = 0$ is imposed at the bottom boundary, and no boundary condition is imposed at the outflow (right) boundary.

For all simulations we consider an unstructured mesh consisting of 1,315 nodes and 2,464 elements. For the reference values used in Eq. (6), we consider the initial condition values for the left domain. Figure 6 shows the 2D density distribution obtained with all methods. Figure 7 shows the density profile along $y = 0.25$, obtained with SUPG + CAU, SUPG + YZβ and NMV methods. We may observe a good agreement between the SUPG + YZβ and NMV solutions, clearly exhibit less dissipation than the SUPG + CAU solution.

One more time, the NMV method needs less GMRES iterations and CPU time than the others, as we can see in Table 2. The NMV method need less than half the number of GMRES iterations required by SUPG + CAU and SUPG + YZβ methods. Additionally, the NMV method requires approximately 48 % and 77 %, respectively, of the CPU time required by the SUPG + CAU and the SUPG + YZβ methods.

Table 2. Reflected shock problem: computational performance

Methods	GMRES Iterations	CPU Time (s)
SUPG + CAU	80,948	198.365
SUPG + YZβ	35,023	124.468
NMV	17,482	96.139



(a) SUPG + CAU

(b) SUPG + YZβ

(c) NMV

Fig. 6. Reflected shock problem: density distribution 2D solution at time $t = 3$. (Color figure online)

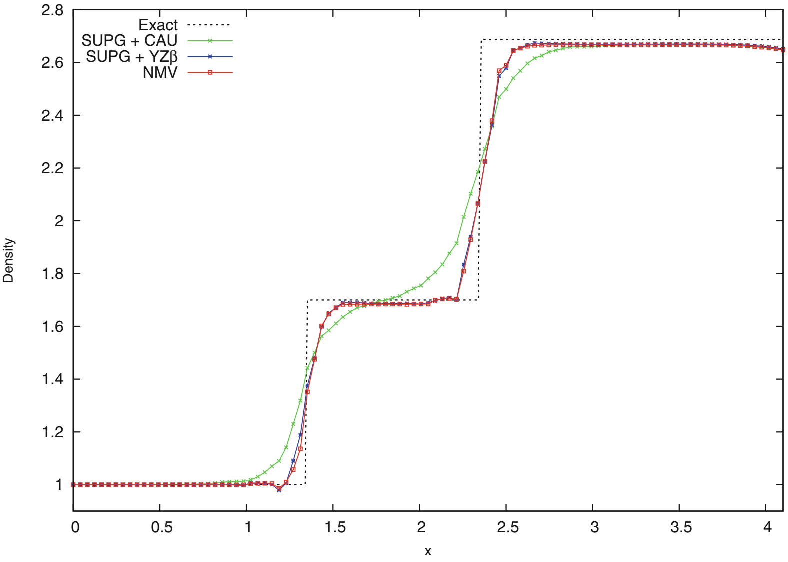


Fig. 7. Reflected shock problem: density profile along $y = 0.25$. (Color figure online)

3.3 2D Explosion Problem

We consider the explosion problem for an ideal gas with $\gamma = 1.4$ as described by [1]. The 2D Euler equations are solved on a 2.0×2.0 square domain in the $xy-$plane. The initial condition consists of the region inside of a circle with radius $R = 0.4$ centered at $(1, 1)$ and the region outside the circle, see Fig. 8. The flow variables are constant in each of these regions and are separated by a circular discontinuity at time $t = 0$. The two constant states are chosen as

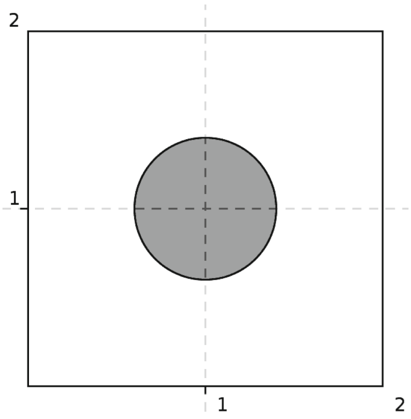


Fig. 8. Explosion problem description.

$$\text{ins}\begin{cases}\rho &= 1.0\\ u &= 0.0\\ v &= 0.0\\ p &= 1.0\end{cases}\qquad \text{out}\begin{cases}\rho &= 0.125\\ u &= 0.0\\ v &= 0.0\\ p &= 0.1\end{cases}\tag{13}$$

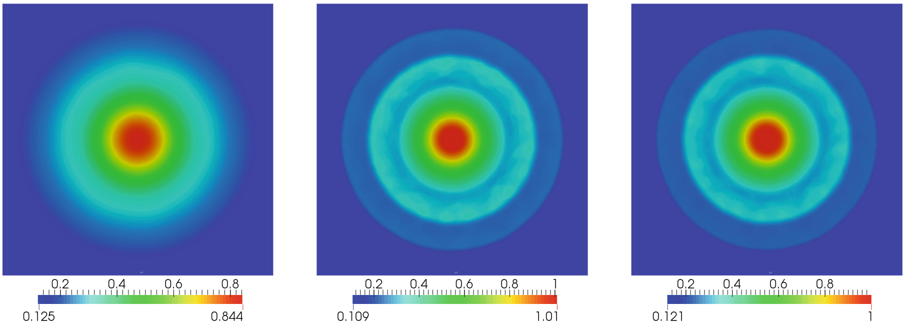
Subscripts ins and out denote values inside and outside the circle respectively.

Table 3. Explosion problem: computational performance

Methods	GMRES Iterations	CPU Time (s)
SUPG + CAU	1,273,137	9,965.661
SUPG + YZβ	15,012	178.953
NMV	11,228	70.029

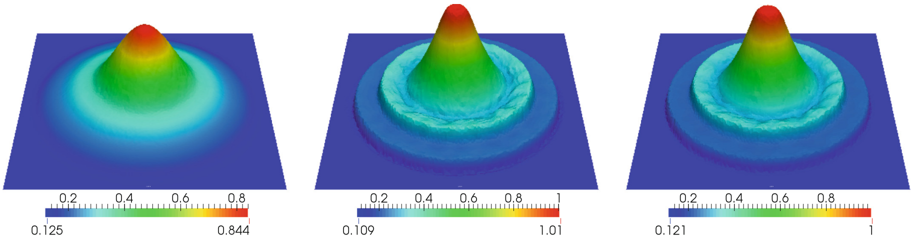
A reference solution was used considering a fine mesh with 1000 × 1000 computing cells by WAF method and it is in good agreement with the analytical solution as described in [21]. In our simulation, we consider an unstructured mesh with 13,438 nodes and 26,474 elements. For the reference values used in Eq. (6), we consider the initial condition values for inside the circle. Figure 9 shows the 2D density distribution and Fig. 10 shows the 3D density distribution obtained with all methods. Figure 11 compares the radial variations of the density obtained using SUPG + CAU, SUPG + YZβ and NMV methods. The solution obtained with NMV is slightly more accurate than the SUPG + YZβ solution, whereas the SUPG + CAU solution clearly exhibit more dissipation.

Again, the NMV method needs less GMRES iterations and CPU time than the others, as we can see in Table 3. The NMV method needed less GMRES iterations than required by SUPG + CAU and SUPG + YZβ methods. In addition,



(a) SUPG + CAU (b) SUPG + YZβ (c) NMV

Fig. 9. Explosion problem: density distribution 2D solution at time $t = 0.25$. (Color figure online)



(a) SUPG + CAU (b) SUPG + YZβ (c) NMV

Fig. 10. Explosion problem: density distribution 3D solution at time $t = 0.25$. (Color figure online)

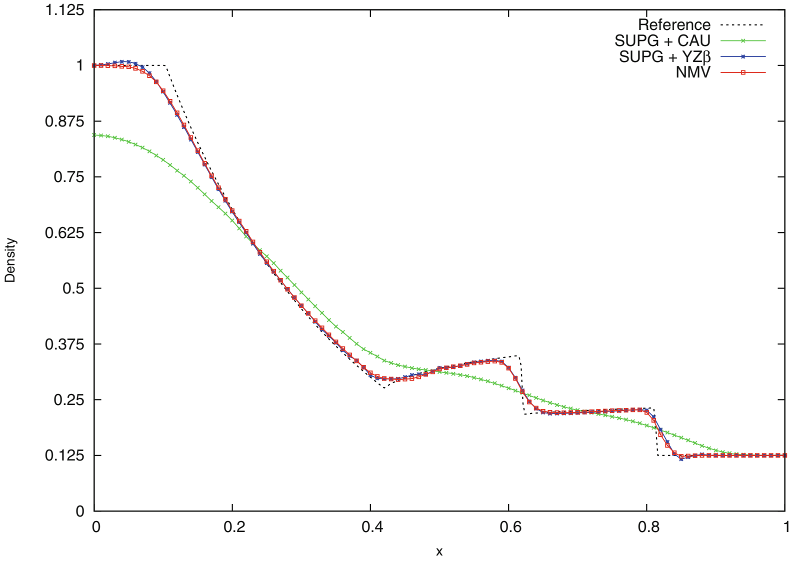


Fig. 11. Explosion problem: comparisons of radial variations of density obtained using SUPG + CAU, SUPG + YZβ and NMV, with the reference solution. (Color figure online)

the NMV method requires approximately 0.7 % and 39 %, respectively, of the CPU time required by the SUPG + CAU and the SUPG + YZβ methods.

4 Conclusions

We presented a new nonlinear multiscale finite element formulation self adaptive for the inviscid compressible flows in conservative variables, where the amount of artificial viscosity is determined by the YZβ shock-capturing parameter. Solutions obtained with the NMV method is comparable with those obtained with the SUPG + YZβ method in oblique shock and reflected shock problems, whereas in the explosion problem the NMV method yields solutions slightly more accurate. Furthermore, the NMV method requires less GMRES iterations and CPU time than the others, as we have seen in the experiments. The NMV method improve conditioning of the linear system, coupled nonlinear equation system that needs to be solved at every time step of a flow computation, which makes substantial difference in convergence of the iterative solution and computationally less costly.

Acknowledgments. This work has been supported in part by CNPq, CAPES and FAPES.

References

1. Abbassi, H., Mashayek, F., Jacobs, G.B.: Shock capturing with entropy-based artificial viscosity for staggered grid discontinuous spectral element method. Comput. Fluids **98**, 152–163 (2014)
2. Arruda, N.C.B., Almeida, R.C., do Carmo, E.G.D.: Dynamic diffusion formulations for advection dominated transport problems. Mecânica Computacional **29**, 2011–2025 (2010)
3. Brenner, S.C., Scott, L.R.: The Mathematical Theory of Finite Element Methods. Texts in Applied Mathematics. Springer, New York, Berlin, Paris (2002)
4. Brooks, A.N., Hughes, T.J.R.: 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 (1982)
5. Catabriga, L., de Souza, D.F., Coutinho, A.L., Tezduyar, T.E.: Three-dimensional edge-based SUPG computation of inviscid compressible flows with YZβ shock-capturing. J. Appl. Mech. **76**(2), 021208–021208-7 (2009). ASME
6. Franca, L.P., Nesliturk, A., Stynes, M.: On the stability of residual-free bubbles for convection-diffusion problems and their approximation by a two-level finite element method. Comput. Methods Appl. Mech. Eng. **166**, 35–49 (1998)
7. Galeão, A., Carmo, E.: A consistent approximate upwind petrov-galerkin method for convection-dominated problems. Comput. Methods Appl. Mech. Eng. **68**, 83–95 (1988)
8. Geuzaine, C., Remacle, J.F.: Gmsh: a 3-D finite element mesh generator with built-in pre- and post-processing facilities. Int. J. Numer. Methods Eng. **79**, 1309–1331 (2009)
9. Gravemeier, V., Gee, M.W., Kronbichler, M., Wall, W.A.: An algebraic variational multiscale-multigrid method for large eddy simulation of turbulent flow. Comput. Methods Appl. Mech. Eng. **199**(13), 853–864 (2010)
10. Hughes, T.J.R., Tezduyar, T.E.: Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible euler equations. Comput. Methods Appl. Mech. Eng. **45**(1), 217–284 (1984)
11. Hughes, T.J.: Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods. Comput. Methods Appl. Mech. Eng. **127**(1–4), 387–401 (1995)
12. John, V., Knobloch, P.: On spurious oscillations at layers diminishing (sold) methods for convection-diffusion equations: part i-a review. Comput. Methods Appl. Mech. Eng. **196**(17), 2197–2215 (2007)

13. Nassehi, V., Parvazinia, M.: A multiscale finite element space-time discretization method for transient transport phenomena using bubble functions. Finite Elem. Anal. Des. **45**(5), 315–323 (2009)
14. Rispoli, F., Saavedra, R., Corsini, A., Tezduyar, T.E.: Computation of inviscid compressible flows with the V-SGS stabilization and YZβ shock-capturing. Int. J. Numer. Methods Fluids **54**(6–8), 695–706 (2007)
15. Santos, I.P., Almeida, R.C.: A nonlinear subgrid method for advection-diffusion problems. Comput. Methods Appl. Mech. Eng. **196**, 4771–4778 (2007)
16. Sedano, R.Z., Bento, S.S., Lima, L.M., Catabriga, L.: Predictor-multicorrector schemes for the multiscale dynamic diffusion method to solve compressible flow problems. In: CILAMCE2015 - XXXVI Ibero-Latin American Congress on Computational Methods in Engineering, November 2015
17. Tezduyar, T.E.: Determination of the stabilization and shock-capturing parameters in supg formulation of compressible flows. In: Proceedings of the European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS (2004)
18. Tezduyar, T.E.: Finite elements in fluids: stabilized formulations and moving boundaries and interfaces. Comput. Fluids **36**(2), 191–206 (2007)
19. Tezduyar, T.E., Senga, M.: Stabilization and shock-capturing parameters in SUPG formulation of compressible flows. Comput. Methods in Appl. Mech. Eng. **195**(13–16), 1621–1632 (2006)
20. Tezduyar, T.E., Senga, M.: SUPG finite element computation of inviscid supersonic flows with YZβ shock-Capturing. Comput. Fluids **36**(1), 147–159 (2007)
21. Toro, E.F.: Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction. Springer Science & Business Media, Heidelberg (2009)