Blucher Mechanical Engineering Proceedings
May 2014, vol. 1 , num. 1
www.proceedings.blucher.com.br/evento/10wccm

DEVELOPMENT OF A 3D COMPRESSIBLE NAVIER-STOKES SOLVER BASED ON A DG FORMULATION WITH SUB-CELL SHOCK CAPTURING STRATEGY FOR FULLY HYBRID UNSTRUCTURED MESHES

R. F. Cantão[1], C. A. C. Silva[2], E. D. V. Bigarella[3], A. C. Nogueira Jr.[4]

[1] Thorus Scisoft Tecnologia Ltda
corresponding author, rfcantao@thorus-scisoft.com.br
[2] Thorus Scisoft Tecnologia Ltda
[3] Empresa Brasileira de Aeronáutica S.A., Embraer
[4] Centro Universitário Salesiano de São Paulo, UNISAL

Abstract. *The development of a CFD tool based on Discontinuous Galerkin discretization is reported. This tool solves the compressible, Reynolds-Averaged Navier-Stokes equations for three-dimensional, hybrid unstructured meshes. This tool is aimed at complex aerospace applications, thus requiring advanced turbulence models and an efficient numerical framework to enhance computational performance and numerical accuracy for high Reynolds number, high Mach number flows. Inviscid fluxes are computed by upwind Roe or HLLC schemes and viscous fluxes are computed using BR1 or BR2 formulations. A 2nd-order accurate, 5-stage, explicit Runge-Kutta time-stepping scheme is used to march the equations in time. The solver computational efficiency, convergence, accuracy and parallel scalability are addressed through flow simulations over typical validation test cases. Convergence rates for increasing degrees of freedom are shown to be asymptotic, with numerical errors compatible to DG schemes. Parallelism is shown to be conformant with the expected scalability behaviour. For the aerospace applications considered in this paper, acceptable agreement with theoretical or experimental results is obtained at adequate computational costs.*

Keywords: *Discontinuous Galerkin, compressible RANS, hybrid meshes, aerospace applications*

1. INTRODUCTION

Viscous, turbulent flow simulations at high Reynolds numbers and at high Mach numbers are typical for aerospace applications. Such flight conditions, however, are still demanding cases for computational tools due to the numerical difficulties related to the flow complexities. The main flow phenomena that exercise the numerical scheme accuracy and robustness are shock waves and turbulent boundary layers, and the interaction between them.

The rapid flow changes provoked by the shock waves usually require some sort of local artificial dissipation in the region to avoid wiggles in the flow field. In finite volume techniques, it is usual that the levels of dissipation applied has strong drawbacks in the solution, such as the spreading of the shock wave in typical centred schemes, or the decrease of the order of the scheme for limiting operations typical of monotonic second-order finite volume schemes, as reported in [9, 8]. When it comes to the final application of numerical tools to the aerospace industry, this will have severe impact in aerodynamic forces, such as lift and drag coefficients.

Regarding turbulent flow simulations, the computational resources required considerably increase because of the need for more near-wall grid refinement. Along with the increased mesh refinement, numerical stiffness, inaccuracy, and lack of robustness due to cell skewness are also undesirable side effects. Within this context, simple numerical schemes are neither robust enough nor computationally efficient enough to allow solutions under acceptable computational costs. Such issue also impacts aerodynamic force computations due to spurious errors of the numerical scheme.

Hence, the numerical framework of a successful computational tool must be accurate and robust enough to treat these problems. One of such numerical frameworks that consistently cope with these issues is the Discontinuous Galerkin approach. As an strong feature of this framework, the switch between mesh refinement and polynomial order increase, termed *hp* method in the literature [12], allows for exponential numerical convergence with the total number of degrees of freedom for nonlinear problems [12].

Another very important aspect of DG schemes is that, by a consistent polynomial treatment of property distribution within the elements, the numerical framework is less sensitive to mesh quality than other numerical approaches, which is also an indication of strong numerical convergence. In a practical sense, this advantage directly translates into numerical robustness and aerodynamic force computation accuracy.

It should also be mentioned that DG schemes, because of their very own discontinuous nature, are very localized in terms of spatial discretization of differential operators. This compactness property results in a small communication footprint during the parallelism synchronization step.

Summing up these advantages, the end user is potentially able to obtain accurate results even for drag results, with proved robustness and convergence without losing too much high-valued human-hour in laborious mesh creation and quality enforcement.

The current DG formulation is constructed over an orthogonal, hybrid-element, polynomial basis as proposed in [12]. Such basis is orthogonal in both Cartesian and local coordinate systems, allowing for great simplicity in treating hybrid mesh elements, such as hexahedra, tetrahedra, prisms and pyramids, while still providing simple mass matrices that are easily invertible.

The interface inviscid fluxes are computed by the upwind Roe [16] or HLLC schemes [7]. For the viscous flux computation, special treatment must be applied to account for the face jumps. In the present work BR1 or BR2 schemes [6] are applied. Time marching is performed by a 2nd-order accurate, 5-stage, explicit Runge-Kutta time-stepping scheme [11].

In the present paper, the numerical accuracy and computational efficiency of the present DG numerical framework are evaluated. Invariably, numerical errors decay exponentially with the degrees of freedom, as expected for a DG scheme. Typical aerospace applications are also evaluated, regarding shock wave capturing and laminar boundary layers. Good agreement is found with reference data from theory and other CFD results.

This paper is the result of a major R&D project made by Thorus Scisoft company. With the present numerical framework, Thorus Scisoft aims to produce an efficient, robust and accurate tool for industrial applications, with focus on the aerospace area.

The introduction section describes the motivation for the current DG framework development. In the second section of this paper, the theoretical formulations embedded in the numerical tool are briefly presented. In the third section, a description of the DG spatial discretization scheme is highlighted. Evaluation of numerical simulation results is presented in the fourth section. A concluding remark section closes the paper summarising the main achievements and future developments from the current effort.

2. NOMENCLATURE

The following nomenclature will be used along the text.

a	Speed of sound
e	Total energy per unit of volume
e_i	Internal energy
$\mathbf{F} = \mathbf{F}_E - \mathbf{F}_V$	Flux vector
$\mathbf{F}_E$	Inviscid flux vector
$\mathbf{F}_V$	Viscous flux vector
$\hat{\imath} = \{\hat{\imath}_x, \hat{\imath}_y, \hat{\imath}_z\}$	Cartesian unit vector
M	Mach number
$\mathbf{n} = \mathbf{S}/\lvert\mathbf{S}\rvert$	Area unit vector
p	Static pressure
Pr	Prandtl number
$\mathbf{q}$	Heat transfer vector
$\mathbf{Q}$	Conserved variable vector
Re	Reynolds number
$\mathbf{S}$	Area vector
$v_n = \mathbf{v} \cdot \mathbf{n}$	Face normal velocity
$\mathbf{V} = \{u, v, w\}$	Cartesian velocity vector
α	Angle of attack
δ_{ij}	Kronecker delta
μ	Dynamic viscosity coefficient
ρ	Density
τ^{ℓ}	Shear-stress tensor
τ^{t}	Reynolds-stress tensor
Subscript	
i, nb	Left and right cells that share the kth face
$i, j = \{x, y, z\}$	Indices used within the Einstein indexing notation
$, j$	Differential in the jth coordinate-system component
ℓ	Face index
∞	Freestream
Superscript	
$\tilde{\ }$	Face operator
e	Element index
$\ell(e)$	Element index neighbouring the eth element by its ℓth face

3. THEORETICAL FORMULATION

The flows of interest in the present context are modelled by the 3-D compressible Reynolds-Averaged Navier-Stokes (RANS) equations, written in dimensionless form and assuming a perfect gas, as

$$\frac{\partial \mathbf{Q}}{\partial t} + \nabla \cdot (\mathbf{F}_E - \mathbf{F}_V) = 0 \qquad \mathbf{Q} = \left[\begin{array}{ccccc} \rho & \rho u & \rho v & \rho w & e \end{array}\right]^{\mathbf{T}} \tag{1}$$

The inviscid and viscous flux vectors are given as

$$\mathbf{F}_E = \left\{ \begin{array}{c} \rho\,\mathbf{V} \\ \rho\,u\mathbf{V} + p\,\hat{\imath}_x \\ \rho\,v\mathbf{V} + p\,\hat{\imath}_y \\ \rho\,w\mathbf{V} + p\,\hat{\imath}_z \\ (e+p)\,\mathbf{V} \end{array} \right\} \quad \mathbf{F}_V = \frac{1}{Re} \left\{ \begin{array}{c} 0 \\ \left(\tau^{\ell}_{xi} + \tau^{t}_{xi}\right)\,\hat{\imath}_i \\ \left(\tau^{\ell}_{yi} + \tau^{t}_{yi}\right)\,\hat{\imath}_i \\ \left(\tau^{\ell}_{zi} + \tau^{t}_{zi}\right)\,\hat{\imath}_i \\ \beta_i\,\hat{\imath}_i \end{array} \right\} \tag{2}$$

The shear-stress tensor is defined by

$$\tau^{\ell}_{ij} = \mu_\ell \left[\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \frac{\partial u_m}{\partial x_m} \delta_{ij} \right],$$

where u_i is the Cartesian velocity component, and x_i is the Cartesian coordinate. The viscous force work and heat transfer term, β_i, is defined as $\beta_i = \tau_{ij} u_j - q_i$, where the heat transfer component is defined as

$$q_j = -\gamma \left(\frac{\mu_\ell}{Pr} + \frac{\mu_t}{Pr_t} \right) \frac{\partial (e_i)}{\partial x_j}.$$

It is important to remark that, for the flow conditions of interest here, the Reynolds analogy for the turbulent heat transfer as before is adequate and numerically robust. The molecular dynamic viscosity coefficient, μ_ℓ, is computed by the Sutherland law [1]. The dimensionless pressure can be calculated from the perfect gas equation of state.

4. NUMERICAL FORMULATION

4.1. Spatial integration

General formulation

We consider a system of equations that generically represent the RANS equations plus additional transport equations and source terms. This generic system represents the RANS equations coupled to turbulence model equations, such as

$$\frac{\partial \mathbf{Q}}{\partial t} + \nabla \cdot \mathbf{F}_E(\mathbf{Q}) - \nabla \cdot \mathbf{F}_V(\mathbf{Q}, \nabla \cdot \mathbf{Q}) = \mathbf{S}(\mathbf{Q}, \nabla \cdot \mathbf{Q}).$$

The terms which depend on the gradient of the property $\nabla \cdot \mathbf{Q}$ can be linearised as $\mathbf{A}_v = \partial \mathbf{F}_V / \partial \nabla \cdot \mathbf{Q}$ and $\mathbf{A}_s = \partial \mathbf{S} / \partial \nabla \cdot \mathbf{Q}$.

Rewriting the RANS equations as a set of first-order equations and using the previous linearisations, one gets

$$\begin{aligned} \frac{\partial \mathbf{Q}}{\partial t} + \nabla \cdot \mathbf{F}_E - \nabla \cdot (\mathbf{A}_v \cdot \mathbf{P}) &= \mathbf{A}_s \cdot \mathbf{P} \\ \mathbf{P} = \nabla \cdot \mathbf{Q} & \end{aligned}$$

To apply the Discontinuous Galerkin formulation to these equations, we restrict them to a single element Ω^e of the complete domain, and we multiply them by weighting functions $\mathbf{v}$ and $\mathbf{w}$, which are continuous within the domain but discontinuous through adjacent elements. By integrating the fluxes in the RANS equations by parts, the resulting weak solution to the RANS equations in a DG approach is

$$\begin{aligned}\int_{\Omega^e} \mathbf{v}^e \frac{\partial \mathbf{Q}^e}{\partial t} d\Omega^e = &-\sum_{\ell=1}^{N_f^e} \int_{\partial\Omega_\ell} \mathbf{v}_{\partial e}\, \tilde{\mathbf{f}}_{E_\ell} \cdot \mathbf{n}\, dS_\ell + \int_{\Omega^e} \nabla \cdot \mathbf{v}^e\, \mathbf{F}_E^e\, d\Omega^e \\ &+ \sum_{\ell=1}^{N_f^e} \int_{\partial\Omega_\ell} \mathbf{v}_{\partial e} \left(\mathbf{A}_{v_\ell} \cdot \tilde{\mathbf{p}}_\ell\right) \cdot \mathbf{n}\, dS_\ell - \int_{\Omega^e} \nabla \cdot \mathbf{v}^e\, \mathbf{A}_v^e \cdot \mathbf{P}^e\, d\Omega^e \\ &+ \int_{\Omega^e} \mathbf{v}^e\, \mathbf{A}_s^e \cdot \mathbf{P}^e\, d\Omega^e \end{aligned} \tag{3}$$

$$\int_{\Omega^e} \mathbf{w}^e\, \mathbf{P}^e\, d\Omega^e = \sum_{\ell=1}^{N_f^e} \int_{\partial\Omega_\ell} \mathbf{w}_{\partial e}\, \tilde{\mathbf{q}}_\ell \cdot \mathbf{n}\, dS_\ell - \int_{\Omega^e} \nabla \cdot \mathbf{w}^e\, \mathbf{Q}^e\, d\Omega^e, \tag{4}$$

where e superscript represents the eth element of the mesh, and ℓ subscript represents the ℓth face of the eth element.

Inviscid fluxes

In the discontinuous Galerkin formulation, jump of properties are allowed through the interface. This approach hinders the computation of the physical flux in that region since the property is doubly valued in the face, coming from the left and the right neighbouring elements *of the face*. Hence, numerical fluxes are necessary in the faces to account for such jumps. In the previous formulation, the numerical fluxes in the faces are represented by $\tilde{\mathbf{f}}_{E_\ell}$ and $\tilde{\mathbf{f}}_{V_\ell}$ for the inviscid and viscous fluxes, respectively. For the computation of the inviscid numerical flux, any upwind scheme, such as the Roe [16] or HLLC [7] Riemann solvers, can be used. In the present implementation, both schemes are currently considered.

Viscous fluxes

The viscous flux computation is composed basically by two steps, firstly the determination of the auxiliary variable $\mathbf{P}$ in Eq. (4), and subsequently its use in the computation of the actual viscous flux in Eq. (3). These two steps must be carefully defined in order to avoid numerical inaccuracy and instability arising from the property jumps in the faces for the elliptic viscous fluxes [3].

Equation (4) can be integrated by parts again resulting in the following equivalent form:

$$\int_{\Omega^e} \mathbf{w}^e\, \mathbf{P}^e\, d\Omega^e = \sum_{\ell=1}^{N_f^e} \int_{\partial\Omega_\ell} \mathbf{w}_{\partial e}\, \left(\tilde{\mathbf{q}}_\ell - \mathbf{Q}^e\right) \cdot \mathbf{n}\, dS_\ell + \int_{\Omega^e} \mathbf{w}^e\, \nabla \cdot \mathbf{Q}^e\, d\Omega^e. \tag{5}$$

Moving the volume integral on the right hand side to the left, one gets

$$\int_{\Omega^e} \mathbf{w}^e\, \boldsymbol{\delta}^e\, d\Omega^e = \sum_{\ell=1}^{N_f^e} \int_{\partial\Omega_\ell} \mathbf{w}_{\partial e}\, \left(\tilde{\mathbf{q}}_\ell - \mathbf{Q}^e\right) \cdot \mathbf{n}\, dS_\ell, \tag{6}$$

where $\boldsymbol{\delta}^e = \mathbf{P}^e - \nabla \cdot \mathbf{Q}^e$ can be understood as a correction of the state gradients that takes into account the effect of the interface discontinuities.

The solution of the viscous terms in the RANS equations now relies on the definition of $\tilde{\mathbf{q}}$ in Eq. (6) and $\tilde{\mathbf{p}}$ in Eq. (3). Various options are available to define these fluxes, as addressed in [3]. Among these methods, one that presents advantageous numerical performance, convergence and accuracy is the BR2 scheme [6]. The definitions for the BR2 scheme are

$$\tilde{\mathbf{q}}_\ell \equiv \{\mathbf{Q}\}_\ell \tag{7}$$

$$\tilde{\mathbf{p}}_\ell \equiv \left\{\nabla \cdot \mathbf{Q}\right\}_\ell - \eta\left\{\boldsymbol{\delta}^e\left(\ell\right)\right\}_\ell \tag{8}$$

with the average operator defined as $\{\cdot\}_\ell = \frac{1}{2}\left((\cdot)^e + (\cdot)^{\ell(e)}\right)$, where the notation $\ell(e)$ indicates the neighbouring element of the eth element by the ℓth face. Variable η is a stabilisation parameter that represents, in essence, the number of faces per element [4].

For the BR2 scheme, $\boldsymbol{\delta}^e$ is defined for the ℓth face of the eth element and can be mathematically written as

$$\int_{\Omega^e} \mathbf{w}^e\ \boldsymbol{\delta}^e\left(\ell\right)\ d\Omega^e = \frac{1}{2}\int_{\partial\Omega_\ell} [\mathbf{Q}]_\ell\ \mathbf{w}_{\partial e} \cdot \mathbf{n}\ dS_\ell, \tag{9}$$

where the jump operator $[\cdot]_\ell$ is defined in the ℓth face of the eth element as $[\cdot]_\ell = (\cdot)^e\,\mathbf{n} - (\cdot)^{\ell(e)}\,\mathbf{n}$. This definition indicates that only the jump contribution pertinent to the ℓth face as required by the flux operator $\tilde{\mathbf{p}}_\ell$ is computed. With such definition, it is guaranteed that only the face neighbours of the eth element will be considered in the assembly of the element system matrix, thus avoiding greater stencils and still guaranteeing numerical accuracy and convergence, and also keeping the parallelism communication overhead at a minimum.

It is worthy to remark here that an option in the code is also available that defines the auxiliary problem as $\mathbf{P} = \mathbf{A}_v \nabla \cdot \mathbf{Q}$ [5]. With that definition and assuming that $\mathbf{w} = \nabla \cdot \mathbf{v}$, the auxiliary problem in Eq. (5) can be substituted back into the main problem, Eq. (3), if no source terms are considered [5].

4.2. Polynomial bases

One fundamental part of a DG framework is the polynomial basis for the solution representation. These requirements translate into a basis which is orthogonal at least in the local coordinate system, so as mass matrices are easily invertible, yet generic enough to treat different cell types, such as hexahedra, tetrahedra, prisms and pyramids that are found in daily meshes for engineering applications. One should note that such requirements are really demanding when considering 3-D applications. Under such requirements, practical high-order capability is readily achieved with a DG framework with adequate computational efficiency and robustness.

One of such basis is the hierarchical tensor-type one proposed in [12]. Such basis is orthogonal in both Cartesian and local coordinate systems, allowing for great simplicity in treating hybrid discretizations which combines structured and unstructured domains consisting of polymorphic subdomains such as tetrahedra, hexahedra, triangular prisms, and pyramids. One should note that the chosen polynomial expansion is of a modal type [12] instead of a nodal one [10]. Such option is critical to deal with practical applications which rely upon 3D hybrid unstructured meshes. It has also many benefits from the computational point of view contributing heavily for an efficient and robust numerical framework [12].

4.3. Artificial dissipation for shock waves

Although DG formulations naturally combine high-order discretizations with consistent shock wave treatment, in practice this framework is still prone to wiggle formation near discontinuities. In other words, the numerical dissipation already provided by the jumps and the flux scheme choice are not enough to counterpart the Gibbs phenomenon close to the shock waves when solving with high-order discretization.

Capturing shock waves with high-order discretization is also a demanding task in finite difference and finite volumes formulations. Several approaches such as switched centred artificial dissipation [13] or MUSCL-like reconstruction techniques [17] with property limiting are usual in such discretization formulations. Switched artificial dissipation terms or limiting can be understood as both decreasing the order of the formulation nearby discontinuities. More advanced schemes, such as WENO-type polynomials, have not proved very practical in multidimensional formulation. Switched artificial dissipation formulations are also known to spread the shock wave through 2 or more cells within the domain, which is a strong drawback for such type of scheme.

In a DG framework, nevertheless, it can be shown [14] that sub-cell shock capturing for high-order polynomials can be achieved for a simple viscosity artificial dissipation model. This is a very favourable observation since the simple viscosity model avoids the large computational difficulties associated with reconstruction or WENO-type methods, while avoiding the loss of accuracy of the inner-cell polynomials as found in reconstruction schemes. In order to further avoid extra dissipation due to the viscosity model, a shock wave sensor can be used to trigger dissipation near the discontinuity [14]. The implementation of the artificial dissipation scheme in the present numerical framework follows the shock indicator formulation and viscous flux modification suggested in [5].

5. COMPUTATIONAL ASPECTS

The numerical framework is implemented in C++ an can be roughly decomposed in the following components:

Problem setup and data covers the specification and API for both a high-performance, multiplataform, binary data format able to handle hybrid meshes and result fields of varying h and p, and a XML and DTD schema for material properties, boundary conditions and solver parameters. It should be noted that converters to and from the major pre and post processors are provided, or are easy to implement;

Mesh services provide all mesh related manipulation, including partitioning for parallel execution;

Numerical core implements all DG related numerical operations, including volumetric and facial quadratures, diferentiation, reconstructions and so on. It's the cornerstone of the solver;

Modelling core represents the highest level of abstraction, orchestrating the functionalities of the previous modules into full blown CFD models.

State of the art, tried and proven computational libraries — Boost, OpenMPI, Eigen and Xerces, to name but a few — are used wherever and whenever convenient.

6. RESULTS AND DISCUSSION

In this section, the results obtained with the previously presented DG framework are discussed. Firstly, a solver verification analysis is performed to assess the actual order of accuracy and the sensitivity to mesh topology of the current numerical framework. Shock wave capturing is evaluated with a 1-D shock tube problem. The laminar boundary layer over a flat plate is, then, used to further validate the viscous flux computations.

6.1. Solver verification — Euler equations

An accurate method for performing strong code verification is presented in [15]. A source term carrying information of a preliminarily known solution for the system of equations is explicitly added to the code formulation in order to drive the numerical solution to the known one. The difference between the converged computational solution and the original one can be a measure of the accuracy of the method as well as a confirmation of the correctness of the implementation.

The analytical functions are here prescribed for the primitive variables of the formulation. The solution for these variables in a generic ith quadrature point is chosen as

$$\left\{ \begin{array}{c} \rho \\ u \\ v \\ w \\ p \end{array} \right\}_i = \left\{ \begin{array}{c} 1 + \tanh(x+y+z)_i \\ 1 + \tanh(x+y+z)_i \\ 1 + \tanh(x+y+z)_i \\ 1 + \tanh(x+y+z)_i \\ 5 + \tanh^3(x+y+z)_i \end{array} \right\} , \tag{10}$$

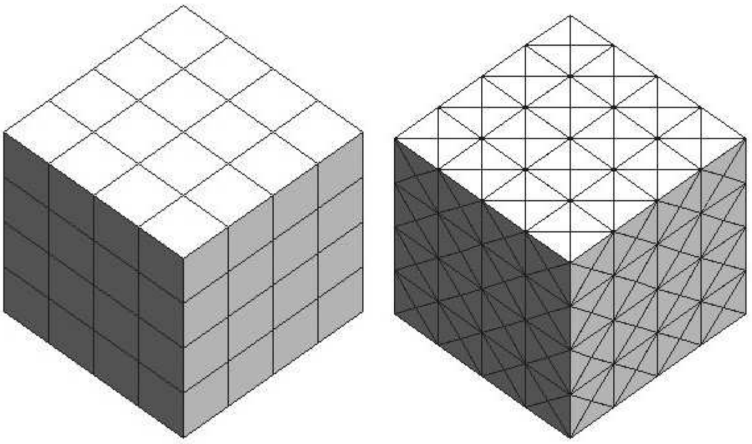
where x_i, y_i and z_i are the dimensionless Cartesian coordinates of the quadrature point. These definitions for the primitive variables are substituted into the Euler equation residue, therefore defining the source terms that are expected to steer the numerical solution towards the prescribed one before. Boundary conditions of the Dirichlet type are used, which means that the values of the conserved properties must be provided at the boundaries for the verification tests.

The chosen physical domain is a hexagonal block with unit sides. The L_2 norm of the difference between the converged computational solution and the original one is assumed as a measure of the accuracy of the method. A set of successively refined tetrahedra and uniform hexahedra meshes is used, ranging from 2 to 25 points, at each edge. It is worthy to stress that the prescribed solution is not aligned with any normal-to-face vectors. With the chosen computational meshes, the authors attempt to address the behaviour of the numerical code with typical grid characteristics such as refinement and topology. A view of the meshes with 4 elements at each edge can be found in Fig. 1. The data presented was computed with the upwind Roe scheme. Similar results were obtained with the HLLC scheme.

Figure 2 presents the logarithm of the mean error curves for polynomial orders ranging from $p = 0$ to 5 plotted against the logarithm of the cell size. It can be clearly seen in these results the faster decrease in numerical errors with the increase of the polynomial order, as expected for a DG formulation. In general, the rates of convergence with polynomial order are a bit larger than the theoretical estimation $P + 1$. Figure 2 also compares the actual rate of convergence (lines with solid circles) compared with the theoretical estimation (small triangles). Away from the machine error bottom line, convergence rates are consistent with the polynomial orders for the two mesh types.

It is worthy to note, nevertheless, the difference observed in the error levels between tetrahedra and hexahedra meshes. This difference relates to the polynomial space spanned at the element, which differs depending on the element type. In the case of hexahedra of order P, the generated polynomial space has polynomials of order up to $3P$, although they do not

Figure 1. Computational meshes with 4 elements along each cube edge.



(a) Hexahedra mesh.

(b) Tetrahedra mesh.

complete the polynomial space of order $3P$. In the case of tetrahedra, the generated polynomial space is up to P complete. In the literature, these extra polynomial modes associated to hexahedra elements are sometimes referred to as *spurious modes*.

Despite the differences between the polynomial spaces in tetrahedra and hexahedra elements, the results show that it is possible to choose a combination of element size and polynomial order for each type of element that leads to the intended solution accuracy.

6.2. Solver verification — Navier-Stokes equations

Following the same method presented in the previous section, an analytical solution of the form given below was prescribed for each quadrature point i to exercise the implemented viscous flux schemes.

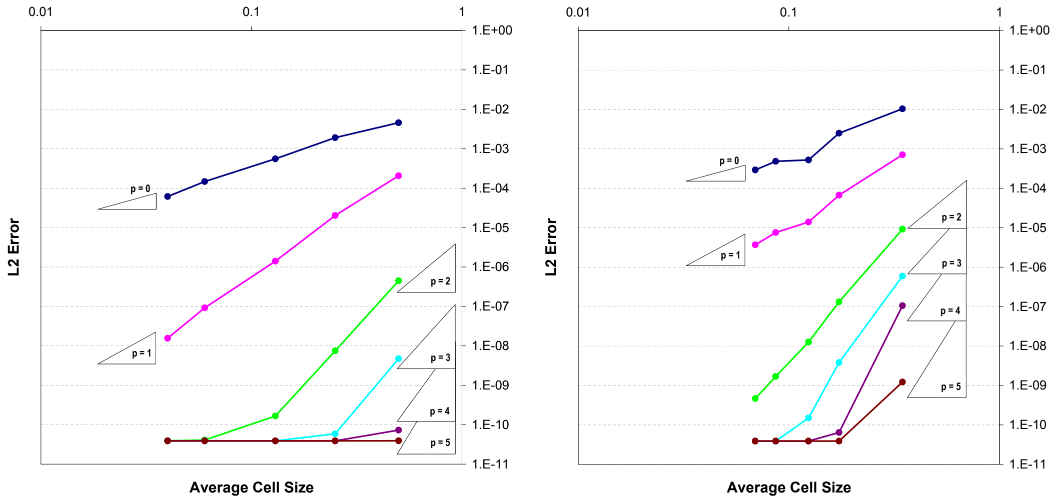
$$\left\{ \begin{array}{c} \rho \\ u \\ v \\ w \\ T \end{array} \right\}_i = \left\{ \begin{array}{c} 1 + 0.1\ \sin(\omega x)_i \\ 1 + 0.04\ \cos(\omega x)\ \sin(\omega y)\ \cos(\omega z)_i \\ 0 \\ 0 \\ (84 + 28y + 10z^2)_i \end{array} \right\}, \tag{11}$$

The Navier-Stokes equations were then integrated using the previously described consistent source term and adequate Dirichlet boundary conditions. 1000 time steps of size 10^{-5} of an explicit Euler integrator were performed for time integration. This example, as proposed in [12], is tailored to validate the code and demonstrate the spectral convergence property of the DG formulation in hybrid meshes.

The physical domain is a cubic block having each coordinate spanning the interval $[-1, 1]$. This block was then spatially discretized by five different meshes as the following: the first four are homogeneus meshes of hexahedra (8 elements), prisms (32 elements), pyramids (48 elements) and tetrahedra (96 elements)[1]. The last mesh is a 14-element hybrid one, containing all four elemental topologies. The mesh choices are made to illustrate the flexibility of the method as well as its exponencial convergence insensitivity with respect to the

[1] The prisms and tetrahedra meshes are nested inside the hexahedra mesh with 8 elements. The pyramids mesh is nested inside a different hexahedra mesh with only 4 elements.

Figure 2. Mean errors for different mesh topologies and sizes, and different polynomial orders.



(a) Hexahedra meshes.

(b) Tetrahedra meshes.

mesh discretization. The maximum (the L_∞ norm) of the difference between the converged computational solution and the analytical one is used as a measure of accuracy of the method.

Figures 3 and 4 show results for the four homogeneous meshes using BR1 and BR2 DG viscous formulations respectively. In this problem, BR1 and BR2 formulations give almost identical results, being the BR2 slightly more precise in all cases. It can be observed that: spectral convergence occurs for all element shapes; the results from the prisms and the hexahedra meshes were identical (P-order prisms have polynomials up to order $2P$ which seems sufficient for the element size selected); the tetrahedra mesh behaved very closely to the hexahedra mesh up to order 7 (the spurious modes were only decisive above that order); the pyramids mesh probably needs a further refinement to match te results of the other meshes (as the polynomial space of tetrahedra and pyramids is the same).

Figure 5 presents the results obtained with the hybrid mesh and BR1 formulation. It can be noted that spectral convergence is also achieved even for a coarse hybric mesh without no special characteristics.

6.3. Shock tube validation

Computations of a 1-D shock-tube flow case is considered. The initial condition is applied with a jump of 20 between the left and and right states for density and pressure. Equal temperatures are assumed at both sides of the shock tube. Numerical results are compared to the analytical solution at time instant $t = 2.5 \times 10^{-4}$. Refinements in mesh and polynomial orders are evaluated.

Pressure and density distributions along the tube are presented in Fig. 6 for successive h refinements. Consistent solution enhancement is achieved with mesh refinement. It is also worthy to point out that no oscillation in the solution is found for all h, indicating that both sub-cell dissipation and viscous flux schemes are stable even for a transient problem. Similar plots for successive p refinement are presented in Fig. 7. Similar conclusions from Fig. 6 can be extended to p variation.

6.4. Laminar Boundary Layer Validation

The laminar boundary layer that develops over a flat plate is also used for the code validation, since an analytical solution exists for this problem [2]. Here, a subsonic flow with $M_\infty = 0.2$ and $Re = 10^5$ based on the plate length is considered. The plate length is fixed to one, and the grid extends to two units upstream the plate leading edge, and one unit along the normal direction. Points are clustered near the flat plate leading edge in order to account for the larger gradients that are expected in this region. Here, 20 elements are placed within the boundary layer along the whole plate extension, which means that the normal spacing varies with the plate length, and 30 cells are placed outside the boundary layer. All grids have 60 points along the longitudinal direction over the plate extension. Figure 8 presents the boundary layer obtained with the present numerical code with polynomial order of 1 compared to the theoretical solution. A very good agreement between the numerical and the reference solution is observed, which is indicative of the quality of the numerical solution even for a relatively low polynomial order achievable with a DG methodology.

6.5. PARALLEL SCALABILITY

Parallel scalability was assessed using a wind-body geometry consisting of 612526 tetrahedra (Figure 9). Polynomial orders, total number of degrees of freedom and integrations points can be appreciated on Table 1.

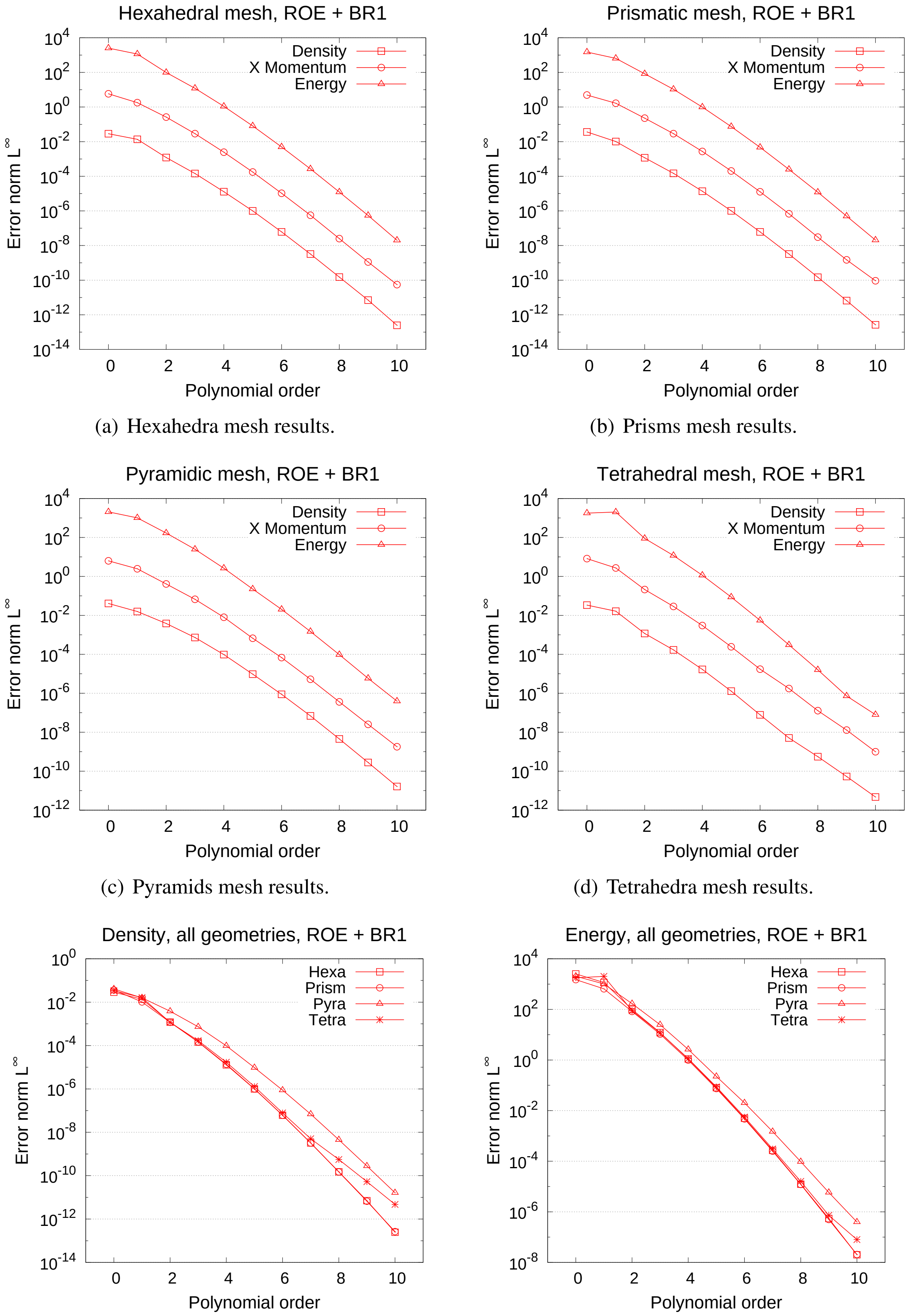
Table 1. Approximation orders, degrees of freedom and integration points for the wing-body geometry.

Order	Dof	Int. Points
0	612526	612526
1	2450104	16538202
2	6125260	76565750

Figure 10 ilustrates the speedup, defined as the ratio between the parallel execution time and the serial execution time (one processor), for several processor numbers ranging from 2 to 48. For each number of processors, 50 time iterations over the Euler solver were made in order to measure execution time.

The scalability of the DG numerical framework is within the expected bounds. Higher orders tend to perform a little better in parallel because they demand more raw processing power per computing node, without a corresponding increase on inter-node communication.

Figure 3. Maximum errors for different mesh topologies and different polynomial orders using the BR1 DG viscous formulation.



(a) Hexahedra mesh results.

(b) Prisms mesh results.

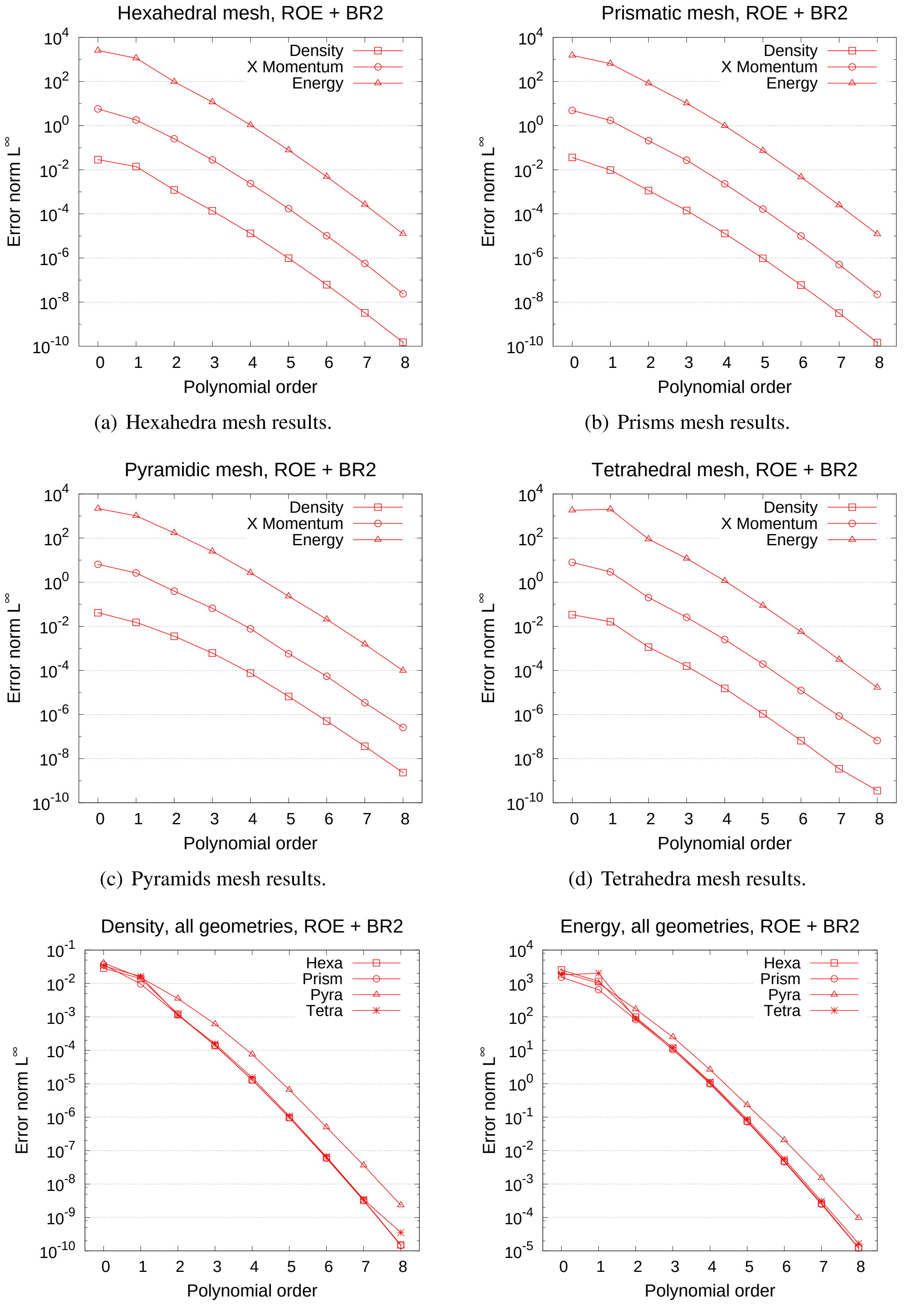
(c) Pyramids mesh results.

(d) Tetrahedra mesh results.

(e) Density for all geometries.

(f) Energy for all geometries.

Figure 4. Maximum errors for different mesh topologies and different polynomial orders using the BR2 DG viscous formulation.



(a) Hexahedra mesh results.

(b) Prisms mesh results.

(c) Pyramids mesh results.

(d) Tetrahedra mesh results.

(e) Density for all geometries.

(f) Energy for all geometries.

Figure 5. Maximal errors for the hybrid mesh and different polynomial orders using the BR1 DG viscous formulation.

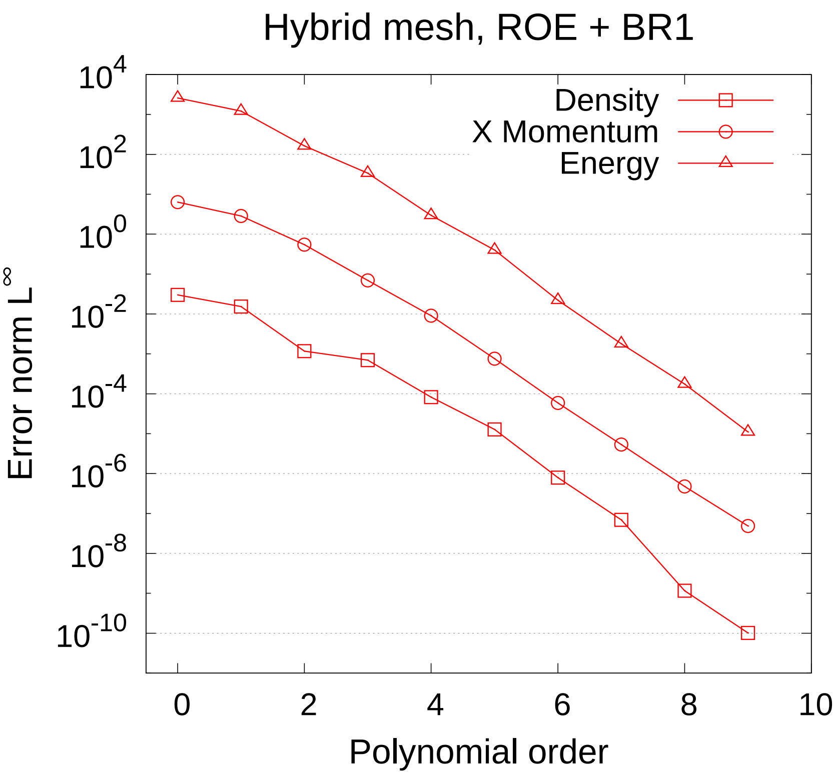
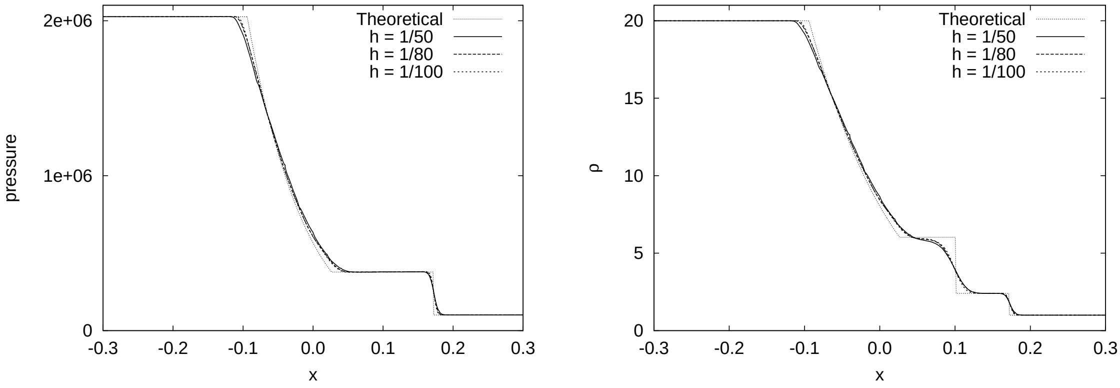


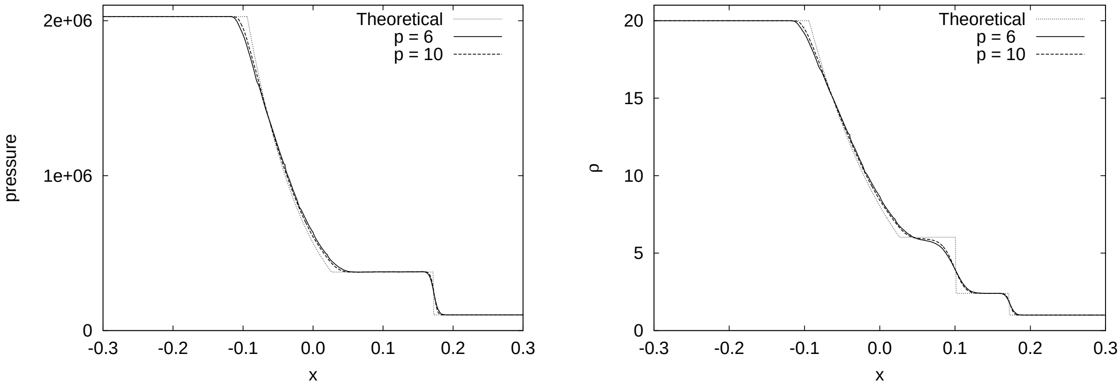
Figure 6. Property distributions along the shock tube obtained with different mesh refinements.



(a) Pressure.

(b) Density.

Figure 7. Property distributions along the shock tube obtained with different polynomial refinements.



(a) Pressure.

(b) Density.

Figure 8. Numerical and analytical laminar boundary layer profiles over a flat plate.

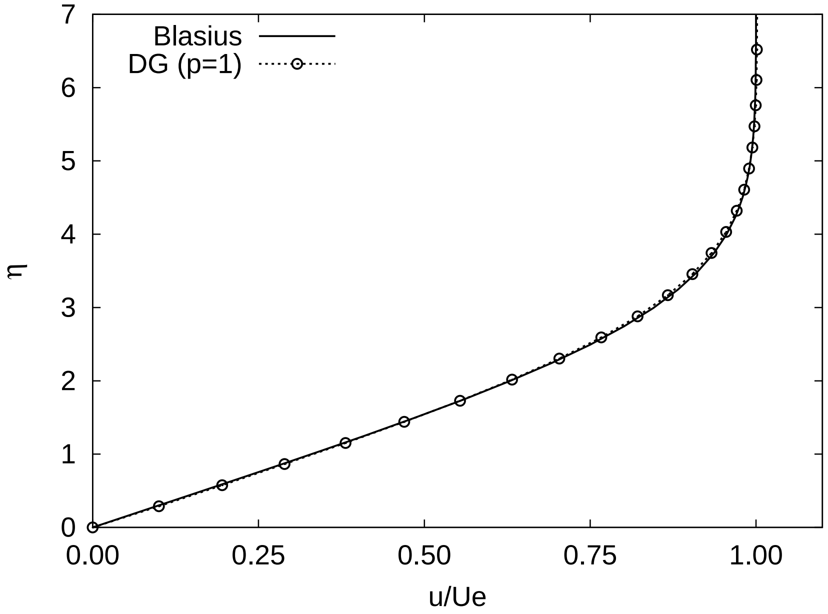


Figure 9. Tetrahedral mesh with 612526 elements used for the assessment of parallel scalability.

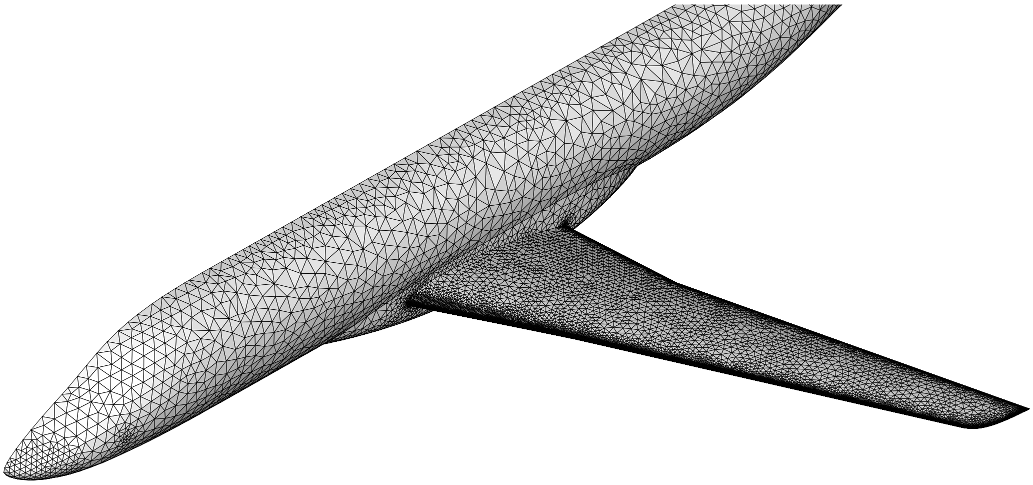
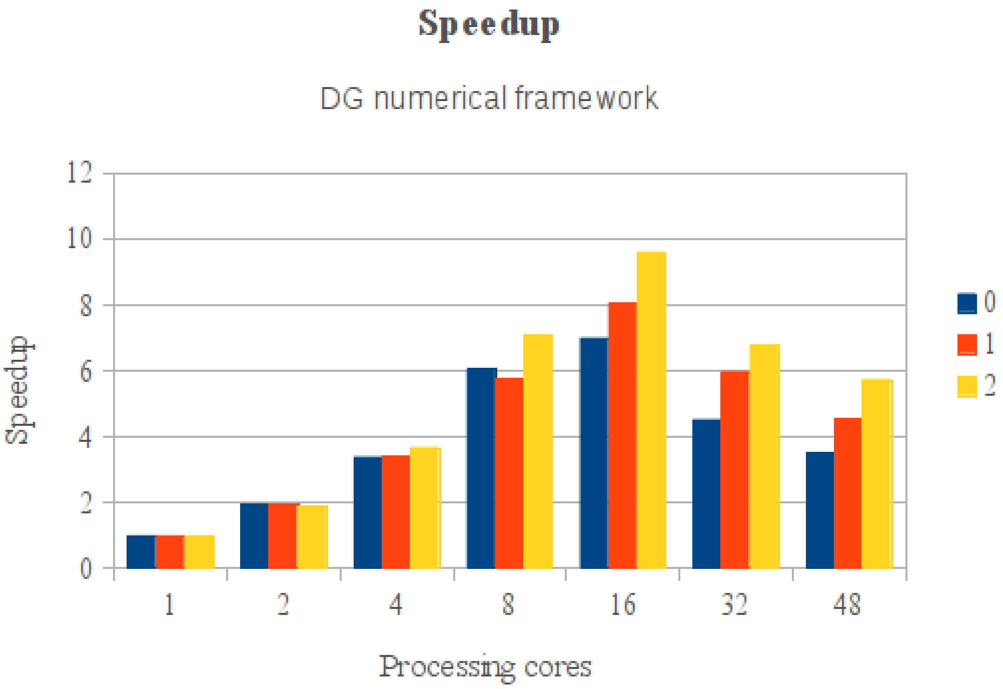


Figure 10. Parallel speedup for the wing-body configuration.



7. CONCLUDING REMARKS

The development of a CFD tool based on the Discontinuous Galerkin method for 3-D, compressible, inviscid and viscous flows has been reported. The inviscid fluxes are computed by upwind Roe or HLLC schemes, whereas the viscous fluxes are computed by BR1 or BR2 formulations. Gibbs phenomenon close to shock waves with high-order discretization are damped with constant sub-cell dissipation.

The code accuracy has been validated by the method of the manufactured solution, where a source term carrying information of a prescribed analytical field for the solution is used as a source term to drive the numerical solution towards the prescribed one. The difference between the converged numerical solution and the prescribed one is the measure of the code accuracy. Error levels were shown to consistently converge with the polynomial order for meshes with any elemental topology, such as hexahedra, tetrahedra, prisms and pyramids, demonstrating the code insensitivity to the mesh element type in that criterion.

Further validation assessment was performed for a laminar boundary layer and a shock tube problems. The numerical framework consistently reproduced the laminar boundary layer of a flat plate. It also adequately represented the shock and expansion wave structures and travelling speeds for a shock tube problem, for polynomial orders up to 10. The numerical framework also shows good parallel execution behaviour. This sequence of validation studies and the obtained results strongly indicate the consistency, robustness and computational performance of the presented framework, as well as its suitability to aerospace application.

Acknowledgments

The authors would like to acknowledge Fundação de Amparo à Pesquisa do Estado de São Paulo, FAPESP, which supported the present work under the Process No. 06-56609-0. The authors would also like to acknowledge Prof. Spencer Sherwin, from the Imperial College London, UK, and Prof. Mike Kirby, from the University of Utah, USA, for their great support and valuable comments on high order DG discretizations. Finally, the authors kindly thank the Centro Universitario Salesiano de São Paulo for financially supporting the publication of this paper.

References

[1] J. D. Anderson, Jr. *Fundamentals of Aerodynamics*, chapter 15, page 647. McGraw-Hill International Editions, New York, NY, USA, second edition, 1991.

[2] J. D. Anderson, Jr. *Fundamentals of Aerodynamics*, chapter 17, pages 723–729. McGraw-Hill International Editions, New York, NY, USA, second edition, 1991.

[3] D. N. Arnold, F. Brezzi, B. Cockburn, , and L. D. Marini. Unified analysis of discontinuous galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis*, 39(5):1749–1779, 2002.

[4] D. N. Arnold, F. Brezzi, B. Cockburn, and D. Marini. Discontinuous Galerkin Methods for Elliptic problems. In C.-W. Shu B. Cockburn amd G. Karniadakis, editor, *Discontinuous Galerkin Methods: theory, computation and applications*, pages 89–101. Springer, 2000.

[5] G. E. Barter. *Shock Capturing with PDE-Based Artificial Viscosity for an Adaptive, Higher-Order Discontinuous Galerkin Finite Element Method*. PhD thesis, Massachusetts Institute of Technology, June 2008.

[6] F. Bassi and S. Rebay. Gmres discontinuous galerkin solution of the compressible navier-stokes equations. In Cockburn, Karniadakis, and Shu, editors, *Discontinuous Galerkin Methods: Theory, Computation and Applications*, pages 197–208. Springer, Berlin, 2000.

[7] P. Batten, M. A. Leschziner, and U. C. Goldberg. Average-state Jacobians and implicit methods for compressible viscous and turbulent flows. *Journal of Computational Physics*, 137(1):38–78, Oct. 1997.

[8] E. D. V. Bigarella. *Advanced Turbulence Modelling for Complex Aerospace Applications*. PhD thesis, Instituto Tecnológico de Aeronáutica, São José dos Campos, SP, Brazil, Oct. 2007.

[9] E. D. V. Bigarella and J. L. F. Azevedo. A study of convective flux computation schemes for aerodynamic flows. In *43rd AIAA Aerospace Sciences Meeting and Exhibit*, AIAA Paper No. 2005-0633, Reno, NV, Jan. 2005.

[10] J. S. Hesthaven and T. C. Warburton. *Nodal Discontinuous Galerkin Methods – Algorithms, Analysis and Applications*, volume 54 of *Texts in Applied Mathematics*. Springer, 1st edition, 2008.

[11] A. Jameson, W. Schmidt, and E. Turkel. Numerical solution of the Euler equations by finite volume methods using Runge-Kutta time-stepping schemes. In *14th AIAA Fluid and Plasma Dynamics Conference*, AIAA Paper No. 81-1259, Palo Alto, CA, June 1981.

[12] G. E. Karniadakis and S. J. Sherwin. *Spectral/hp Element Methods for Computational Fluid Dynamics*. Oxford University Press, Oxford, 2nd edition, 2005.

[13] D. J. Mavriplis. Accurate multigrid solution of the Euler equations on unstructured and adaptive meshes. *AIAA Journal*, 28(2):213–221, Feb. 1990.

[14] P. O. Persson and J. Peraire. Sub-cell shock capturing for discontinuous galerkin methods. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA Paper No. 2006-112. AIAA, 2006.

[15] Patrick J Roache. *Verification and Validation in Computational Science and Engineering*. Hermosa Publishers, Albuquerque, NM, USA, 1998.

[16] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357–372, Oct. 1981.

[17] B. van Leer. Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method. *Journal of Computational Physics*, 32(1):101–136, July 1979.