AFRL-AFOSR-UK-TR-2011-0009

Development of High-Order Methods for Multi-Physics Problems Governed by Hyperbolic Equations

John Ekaterinarius

FORTH/IACM
Box 1527
Heraklion, Greece 71110

EOARD GRANT 07-3099

October 2010

Final Report for 01 November 2007 to 01 November 2010

Air Force Research Laboratory
Air Force Office of Scientific Research
European Office of Aerospace Research and Development
Unit 4515 Box 14, APO AE 09421

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
15-10-2010	Final Report	1 November 2007 – 01 November 2010

4. TITLE AND SUBTITLE

Development of High-Order Methods for Multi-Physics Problems Governed by Hyperbolic Equations

5a. CONTRACT NUMBER

FA8655-07-1-3099

5b. GRANT NUMBER

Grant 07-3009

5c. PROGRAM ELEMENT NUMBER

61102F

6. AUTHOR(S)

Dr. John Ekaterinarius

5d. PROJECT NUMBER

5d. TASK NUMBER

5e. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

FORTH/IACM
Box 1527
Heraklion, Greece 71110

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

EOARD
Unit 4515 BOX 14
APO AE 09421

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/AFOSR/RSW (EOARD)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

AFRL-AFOSR-UK-TR-2011-0009

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report contains the quarterly reports from the PI as follows:
1 March – 31 May 2008……..1
31 Oct - 31 Mar 2009……..15
1 Mar – 30 May 2009…….37
1 June – 31 Aug 2009……..62
1 Jan – 31 March 2010……..70
1 Apr – 30 June 2010……82
1 July – 30 Sep 2010…….96

15. SUBJECT TERMS

EOARD, High Speed Aerodynamic, Computational Fluid Dynamics (CFD), Numerical Simulation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18, NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Surya Surampudi
UNCLAS	UNCLAS	UNCLAS	SAR	109	19b. TELEPHONE NUMBER *(Include area code)* +44 (0)1895 616021

Quarterly Progress Report for the
EOARD Project

Development of High–Order Methods for Multi–Physics Problems Governed by Hyperbolic Equations

by

John A. Ekaterinaris

FORTH / IACM

Award No. FA 8655–03–1–3085

Reporting period March 1 – May 31, 2008

Abstract

A numerical method based on high order WENO discretization of the compressible Nervier-Stokes equations and a sub-grid scale model for LES was tested and validated for the simulation of transition and turbulence of normal and oblique shock boundary layer interaction over a flat plate for a wide range of Mach numbers. An eddy viscosity sub-grid scale model is used for large eddy simulations (LES) for both the turbulent and transitional flow regime. The numerical simulations show that good agreement with available experimental data is obtained for both the predicted pressure distribution and velocity profiles.

1 Introduction

The discretization of the inviscid fluxes is based on standard finite difference WENO schemes. The numerical code includes options for 5th, 7th and 9th order accurate discretizations of the inviscid fluxes. All computations shown here were performed with the 9th order accurate WENO scheme. The viscous fluxes were evaluated with a 4th order accurate explicit, central difference scheme by evaluating the second derivatives with repeated evaluation of the first derivative, first at half points and then at the nodes of the finite difference mesh. High order accurate discretization of the viscous fluxes requires large computational time. It was found that for a fourth order accurate explicit discretization, significant portion of computational time per time step is spend for the evaluation of the viscous fluxes and that higher order explicit or compact discretizations are prohibitively expensive.

Implicit, second order accurate in space and time, time stepping methods exist as option in the code. However, it was found that these implicit schemes are not as accurate for time integration as explicit schemes, and in addition, they become very diffusive and not appropriate for LES at the presence of strong shocks. Therefore all flows were computed with the classical tree stage, third order accurate TVD Runge-Kutta method of Osher and Shu. For LES computations the Smagorinsky sub-grid scale model is used. The

particular implementation of the sub-grid model and the computed results are given in the following sections. Finally, the results and the conclusions of this study are presented.

2 Numerical Implementation for Large Eddy Simulations

The Navier-Stokes solver is based on WENO discretization for the inviscid fluxes and central finite difference discretization of the viscous fluxes. It is parallelized and handles only block structured grids for the numerical solution of the Farve-averaged continuity momentum and energy equations.

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial(\bar{\rho}\tilde{u}_j)}{\partial x_j} = 0 \tag{2.1}$$

$$\frac{\partial(\bar{\rho}\tilde{u}_i)}{\partial t} + \frac{\partial(\bar{\rho}\tilde{u}_i\tilde{u}_j)}{\partial x_j} = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial \tilde{\tau}_{ij}}{\partial x_j} + \frac{\partial \tilde{\tau}_{ij}^{\mathrm{SGS}}}{\partial x_j} \tag{2.2}$$

$$\begin{aligned}\frac{\partial(\bar{\rho}\hat{E})}{\partial t} + \frac{\partial(\bar{\rho}\hat{E} + \bar{p})\tilde{u}_j}{\partial x_j} &= \frac{\partial}{\partial x_j}\Big(c_p\frac{\mu}{Pr}\frac{\partial \tilde{T}}{\partial x_j} - q_j^{\mathrm{SGS}}\Big) \\ &+ \frac{\partial}{\partial x_j}\Big[\tilde{u}_i(\tilde{\tau}_{ij} + \tilde{\tau}_{ij}^{\mathrm{SGS}})\Big]\end{aligned} \tag{2.3}$$

where $\hat{E} = \bar{p}/(\gamma - 1) + 0.5\bar{\rho}\tilde{u}_i\tilde{u}_j$ is the computable energy and $\tilde{\tau}_{ij}$ is the Farve-filtered viscous stress tensor

$$\tilde{\tau}_{ij} = \mu\Big(2\tilde{S}_{ij} - \frac{2}{3}\tilde{S}_{mm}\delta_{ij}\Big) \tag{2.4}$$

where the molecular viscosity is computed from the filtered temperature from the Sutherland law

$$\mu(\tilde{T}) = \tilde{T}^{\frac{3}{2}}\Big[\frac{(1+0.76)}{(\tilde{T}+0.76)}\Big] \tag{2.5}$$

and $\tilde{S}_{ij}$ is the Farve-filtered strain rate tensor given by

$$\tilde{S}_{ij} = \frac{1}{2}\Big(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\Big) \tag{2.6}$$

On the right hand side of the energy equation, Eq. (2.3), the additional diffusive terms exist

$$-\frac{\partial}{\partial x_j}\Big(\gamma C_v \mathcal{Q}_j + \frac{1}{2}\mathcal{J}_j - \mathcal{D}_j\Big) \tag{2.7}$$

these terms are given by

$$\mathcal{Q}_j = \bar{\rho}\Big(\widetilde{u_j T} - \tilde{u_j}\tilde{T}\Big) \tag{2.8}$$

$$\mathcal{J}_j = \bar{\rho}\Big(u_i\widetilde{u_j}u_k - \tilde{u_j}\widetilde{u_k u_k}\Big) \tag{2.9}$$

$$\mathcal{D}_j = \overline{\tau_{ij}u_i} - \tilde{\tau}_{ij}\tilde{u}_j \tag{2.10}$$

Vreman et al. [1], performed *a priori* test using DNS data obtained from the calculation of a compressible mixing layer, and concluded that these nonlinearities have negligible effects and it is acceptable to neglect them. For high Mach number LES, is not absolutely clear that the contribution of these terms is still negligible. However, because a suitable sub-grid scale model is not available for these terms these terms are also neglected for the present LES for computation of shock boundary layer interaction.

The sub-grid-scale viscous tensor $\tilde{\tau}_{ij}^{\mathrm{SGS}}$ is modelled by

$$\tilde{\tau}_{ij}^{\mathrm{SGS}} = \mu_t\Big(2\tilde{S}_{ij} - \frac{2}{3}\tilde{S}_{mm}\delta_{ij}\Big) - \frac{2}{3}\bar{\rho}k^{\mathrm{SGS}}\delta_{ij} \tag{2.11}$$

in the modelled sub-grid-scale tensor $\tilde{\tau}_{ij}^{\mathrm{SGS}}$ the eddy viscosity μ_t and k^{SGS} depend on the sub-grid scale model that is presented in the next section and in Eq. (2.3) the sub-grid heat flux q^{SGS} is modelled by

$$q^{\mathrm{SGS}} = -c_p\frac{\mu_t}{Pr_t}\frac{\partial \tilde{T}}{\partial x_j} \tag{2.12}$$

2.1 Sub-grid Model and Constants

The compressible version of the Smagorinsky sub-grid-scale model [2] with fixed filter width is used. The constants μ_t and k^{SGS} of this model in Eq. (2.11) are evaluated as

$$\mu_t = C_R \bar{\rho} \Delta^2 \sqrt{\tilde{S}_{ij}\tilde{S}_{ij}} \tag{2.13}$$

$$k^{\mathrm{SGS}} = C_I \Delta^2 \tilde{S}_{ij}\tilde{S}_{ij} \tag{2.14}$$

where the constants values $C_R = 0.012$ and $C_I = 0.0066$ are used and the filter width Δ is chosen as $\Delta = (\Delta x \times \Delta y \times \Delta z)^{(1/3)}$

The Smagorinsky sub-grid-scale model is not, however, accurate for wall bounded flows. The deficiencies of this model can be improved when a length scale filter close to the wall is introduced. Therefore the modified version of the Nicoud and Ducros [3] model was also implemented. The eddy viscosity μ_t of this model is given by

$$\mu_t = \bar{\rho}(C_w \Delta)^2 \frac{(\mathcal{S}^d_{ij}\mathcal{S}^d_{ij})^{3/2}}{(\tilde{S}_{ij}\tilde{S}_{ij})^{5/2} + (\mathcal{S}^d_{ij}\mathcal{S}^d_{ij})^{5/4}} \tag{2.15}$$

where the constant value is $C_w = 0.3$ and

$$\mathcal{S}^d_{ij} = \frac{1}{2}\Big(\frac{\partial \tilde{u}_i}{\partial x_k}\frac{\partial \tilde{u}_k}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_k}\frac{\partial \tilde{u}_k}{\partial x_i}\Big) - \frac{1}{3}\frac{\partial \tilde{u}_m}{\partial x_k}\frac{\partial \tilde{u}_k}{\partial x_m} \tag{2.16}$$

The sub-grid-scale viscous tensor $\tilde{\tau}^{\mathrm{SGS}}_{ij}$ of compressible version of the Nicoud and Ducros [3] model is

$$\tilde{\tau}^{\mathrm{SGS}}_{ij} = \mu_t\Big(2\tilde{S}_{ij} - \frac{2}{3}\tilde{S}_{mm}\delta_{ij}\Big) - \frac{2}{3}\frac{C_l}{\bar{\rho}}\Big(\frac{\mu_t}{\Delta}\Big)^2\delta_{ij} \tag{2.17}$$

where the constant value is $C_l = 45.8$.

3 Results

The numerical algorithm with the sub-grid scale model, which was described for completeness in the previous section, was used to perform DNS and LES

of shock boundary layer interaction. Sample results of these computations are presented in this section. The grid used for the simulations and the boundary conditions are described first and then the computed results are presented.

Oblique shock laminar boundary layer interactions for several Mach numbers were computed with DNS and no sub-grid model was used. The oblique shock is generated using isentropic flow relations. The shock boundary layer interaction takes place in a computational domain with length four units and hight one unit. The spanwise extend of the domain is one unit. For all computations the same grid density was used. The characteristics of the mesh are as follows. In the streamwise direction a uniform mesh was used with 125 points per unit length or $\Delta x = 0.008$. Along the normal to the wall direction 200 intervals were used and the spacing of the first point away from the surface was 5×10^{-4}. This grid density guarantees almost 100 points within the boundary layer. Along the periodic spanwise direction fewer intervals were used for cases where we performed DNS because the objective was to capture only the initiation of transition. For LES computations, however, the spacing along the spanwise was the same as in the streamwise direction. The oblique shock angle was set 29^o for all Mach numbers. For this angle of the oblique shock, the wedge angle β and the inflow boundary conditions at the top of the domain for various Mach numbers are specified as follows:

M_∞	β	ρ_{top}	u_{top}	w_{top}	p_{top}
2.9	10.94	1.69997	2.61934	-0.50632	1.52819
4.0	16.80	2.57560	4.05234	-1.22766	5.90900
5.0	19.3	3.24201	4.95422	-1.73494	9.36656

Oblique shock boundary layer interaction results in flow separation as will be shown later. At the both the inflow and the outflow the flow is supersonic, therefore at the inflow supersonic free stream or boundary layer was specified

and at the outflow all quantities were extrapolated from the interior. In addition, at the exit a sponge region of 15 cells was used to completely eliminate reflections that may result from the subsonic portion of the boundary layer.

Normal shock boundary layer interaction was also computed with LES at $M_\infty = 1.1$ and $M_\infty = 1.3$. For the lower Mach number case, shock boundary layer interaction without separation was found. For the higher Mach number, however, significant separation and "Λ" shock formation was found in accordance with the experiment. The computational box has dimensions $L_x = 20 \times L_y = 10 \times L_z = 20$ and a $401 \times 81 \times 151$ point mesh was used. This mesh is uniform in the streamwise and spanwise directions and stretched in the normal to the wall direction with grid spacing for the first point 1×10^{-4}. The boundary conditions at the top of the domain for normal shock boundary layer interaction were specified from the isentropic normal shock relations. At the inflow, the flow is supersonic and laminar boundary with disturbances for the streamwise and normal velocity components [5] was specified. At the outflow, the flow is subsonic therefore nonreflective boundary conditions were specified. In addition, at the exit a sponge region of 15 cells was used to completely eliminate reflections.

3.1 Oblique shock boundary layer interaction

Results for oblique shock laminar boundary layer interaction are shown for $M_\infty = 2.9$ and $M_\infty = 5.0$. The flow field structure of the computed field in the middle of the computational domain is summarized in Fig. 1. The formation of the compression waves over the boundary layer before and after the interaction region are evident. At the interaction region the boundary layer thickness increases significantly. After interaction the compression waves converge to a reflected shock. In the interaction region, massive flow separation is observed for all Mach number. Downstream the interaction region development of disturbances was found.

Several tests were carried out to ensure that these disturbances are interdependent of the inflow boundary conditions. Numerical simulations were carried out for longer domains. The inflow was placed at $x_{in} = -2$ or

$x_{in} = -1$ while the oblique shock was still specified at $x = 0, z = 1$. Simulations were carried out for both fully developed compressible boundary layer at $M_\infty = 2.9$ and developing boundary layer from the leading edge. In all cases the input parameter, which is the Reynolds number per unit length, was adjusted so that at $x = 1.0$ the Reynolds is $Re = 10^4$. The contour plots in Fig. 1 correspond for $x_{in} = -1$.

The details of oblique shock laminar boundary layer interaction at $M_\infty = 2.9$ are shown in Fig. 2. The formation of the compression waves ahead and downstream from the interaction region are indicated by the smooth turning of the streamlines. In the interaction region significant flow separation occurs and the boundary layer reaches about ten times the thickness before the interaction region. The simulations shown in Figs. 1 and 2 correspond to adiabatic wall boundary condition.

The effect of wall temperature on the boundary layer thickens and flow field structure is shown in Fig. 3. The wall temperature from the adiabatic wall boundary condition was found approximately $T_w = 2.5T_\infty$. Three simulations with constant wall temperature were carried out; (1) for wall heated at constant temperature $T_w = 3.5T_\infty$; (2) for wall at a temperature approximately equal to the average adiabatic wall temperature $T_w = 2.5T_\infty$; and (3) for wall with cooling at temperature bellow the adiabatic wall temperature $T_w = 1.5T_\infty$. As expected rise of the wall temperature increases both the thickness of the boundary layer and interaction region while cooling of the wall results into thinner boundary layer. The structure of the flow field is the same for constant wall temperature close to the adiabatic wall temperature.

Flow disturbances were monitored in time upstream and downstream of the interaction region in the region where the boundary layer is fully attached. Self-excited disturbances downstream the interaction region are shown in Fig. 4. For oblique shock boundary layer interaction at $M_\infty = 2.9$ the amplitude of these disturbances is small and are not visible on the contour field plots. However as we will show later the amplitude of these disturbances becomes quite large and they become visible as the free steam Mach number increases.

The general features of the computed flow field in the middle of the com-

putational domain for oblique shock laminar boundary layer interaction at $M_\infty = 5.0$ are shown in Fig. 5. The formation of the compression waves over the boundary layer before and after the interaction region is again evident. The details of oblique shock laminar boundary layer interaction at $M_\infty = 5.0$ are shown in Figs. 6 and 7. The formation of the compression waves waves ahead and downstream from the interaction region are indicated by the smooth turning of the streamlines. In the interaction region significant flow reparation occurs and the boundary layer reaches many times the thickness before and after the interaction region. The disturbances downstream of the interaction region are visible. These disturbances lead to rapid transition into turbulent flow.

3.2 LES for normal shock boundary layer interaction

Simulations for normal shock boundary layer interaction were carried out for $M_\infty = 1.1$ and $M_\infty = 1.3$. For these simulations at the inflow a compressible boundary layer profile was prescribed at Re_δ=5000, 10000, and 20000. At Re_δ=5000, and 10000 transition was found after the interaction at Re_δ=20000 turbulent flow developed before the interaction region. For $M_\infty = 1.1$ normal shock boundary layer interaction the flow at the interaction region was attached in the mean. For $M_\infty = 1.3$ normal shock boundary layer interaction the flow in the interaction region was separated and a Λ-type shock was formed.

The flow field structure for normal shock boundary layer interaction at $M_\infty = 1.1$ is shown with instantaneous iso-surfaces of density and pressure in Figs. 8 and 9, respectively. The flow field structure for normal shock boundary layer interaction at $M_\infty = 1.3$ is presented next. For Re_δ=5000 (see Fig. 10), flow instabilities are visible before the interaction region and transition occurs after the interaction region. Larger scale flow instabilities in the transitional, and turbulent flow regimes are shown in Figs. 11 and 12. At Re_δ=10000 (see Fig. 11) transition occurs in the separated flow region bellow the Λ shock where the formation of hairpin vortices is evident. At Re_δ=10000 (see Fig. 12) transition occurs at the Λ shock boundary layer

interaction region.

Comparisons of the average surface pressure distribution and the axial velocities at several locations with the experiment are shown in Figs. 13 and 14. The computed surface pressure distribution is in good agreement with the measurements. The mean velocity distribution is also in agreement with the experiment and it appears that the main flow features are well captured by the LES.

4 Conclusions

The high order accurate finite difference method we developed is based on WENO discretization of the inviscid fluxes and fourth order accurate discretization of the viscous terms with the Smagorinsky eddy viscosity model for sub-grid scales. This method was fount effective and accurate for the computation of transition and turbulence in high speed compressible flows. Simulations for shock boundary layer interaction showed good agreement with experimental measurements.

References

[1] Vreman, B., Geurts, B., and Kuerten, H., "A Priori Tests of Large Eddy Simulation of the Compressible Plane Mixing Layer," *Journal of Engineering Mathematics,* Vol. 29, 1995, pp. 299-327.

[2] Erlebacher, G., Hussaini, M. Y., Speziale, C. G., and Zang, T. A., "Toward the Large-Eddy Simulation of Compressible Turbulent Flows," *Journal of Fluid Mechanics,* Vol. 238, 1992, pp. 155-185.

[3] Nicoud, F. and Ducros, F., "Subgrid-Scale Stress Modelling Based on the Square of the Velocity Gradient Tensor," *Flow, Turbulence and Combustion,* Vol. 62, No. 3, 1999, pp. 183-200.

[4] Yao, Y., Krishnan, L., Sandham, N.D., and Roberts, G.T., "The Effect of Mach Number on Unstable Disturbances in Shock/Boundary-Layer Interactions," *Physics of Fluids,* Vol. 19, 054104, 2007.

[5] Krishnan, L., Sandham, N.D., "Strong Interaction of a Turbulent Spot with a Shock-Induced Separation Bubble," *Physics of Fluids,* Vol. 19, 016102, 2007.

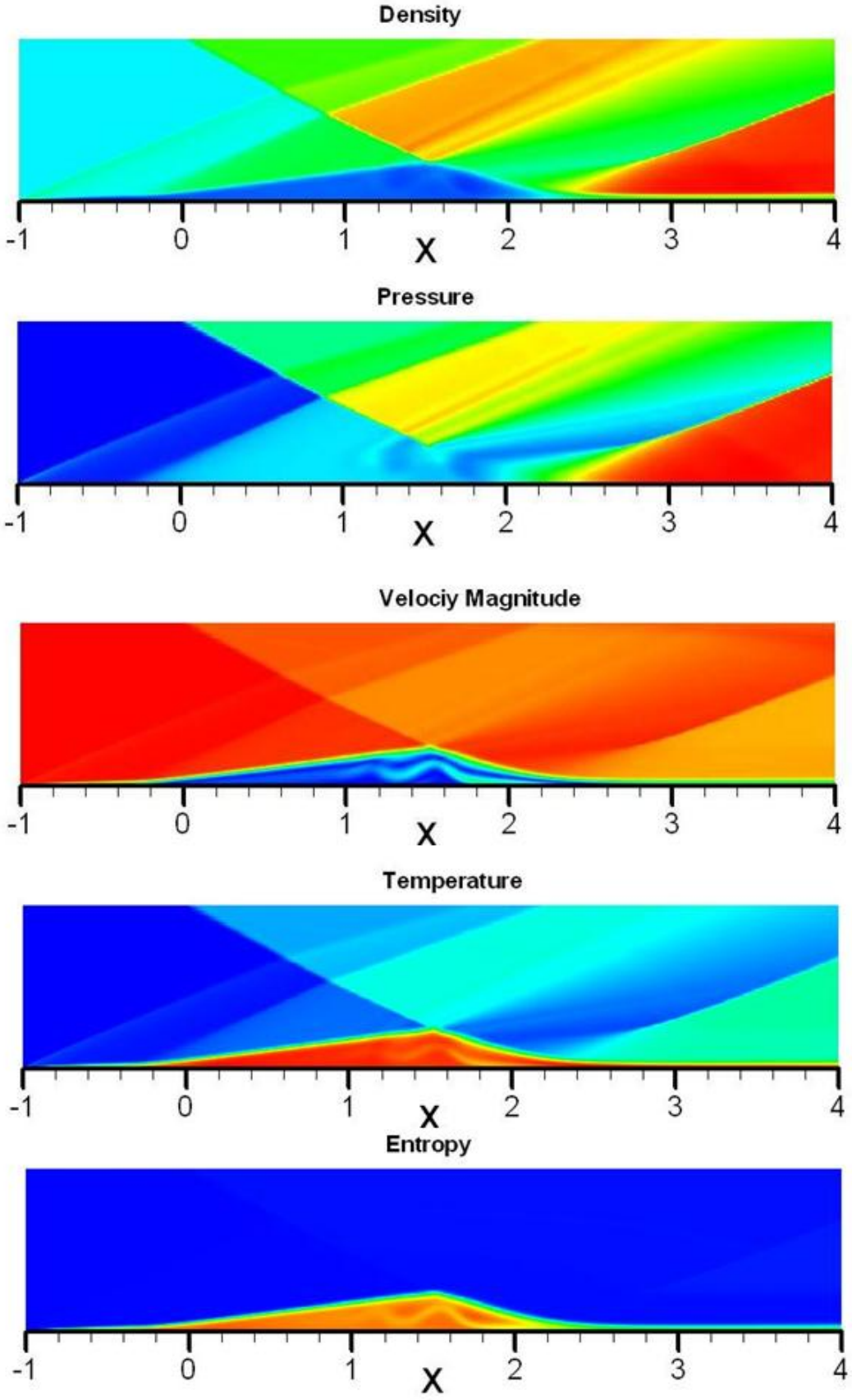


Figure 1. Overview of the computed solution for oblique shock boundary layer interaction at $M_\infty = 2.9$ and incident shock angle $\theta = 29^o$.

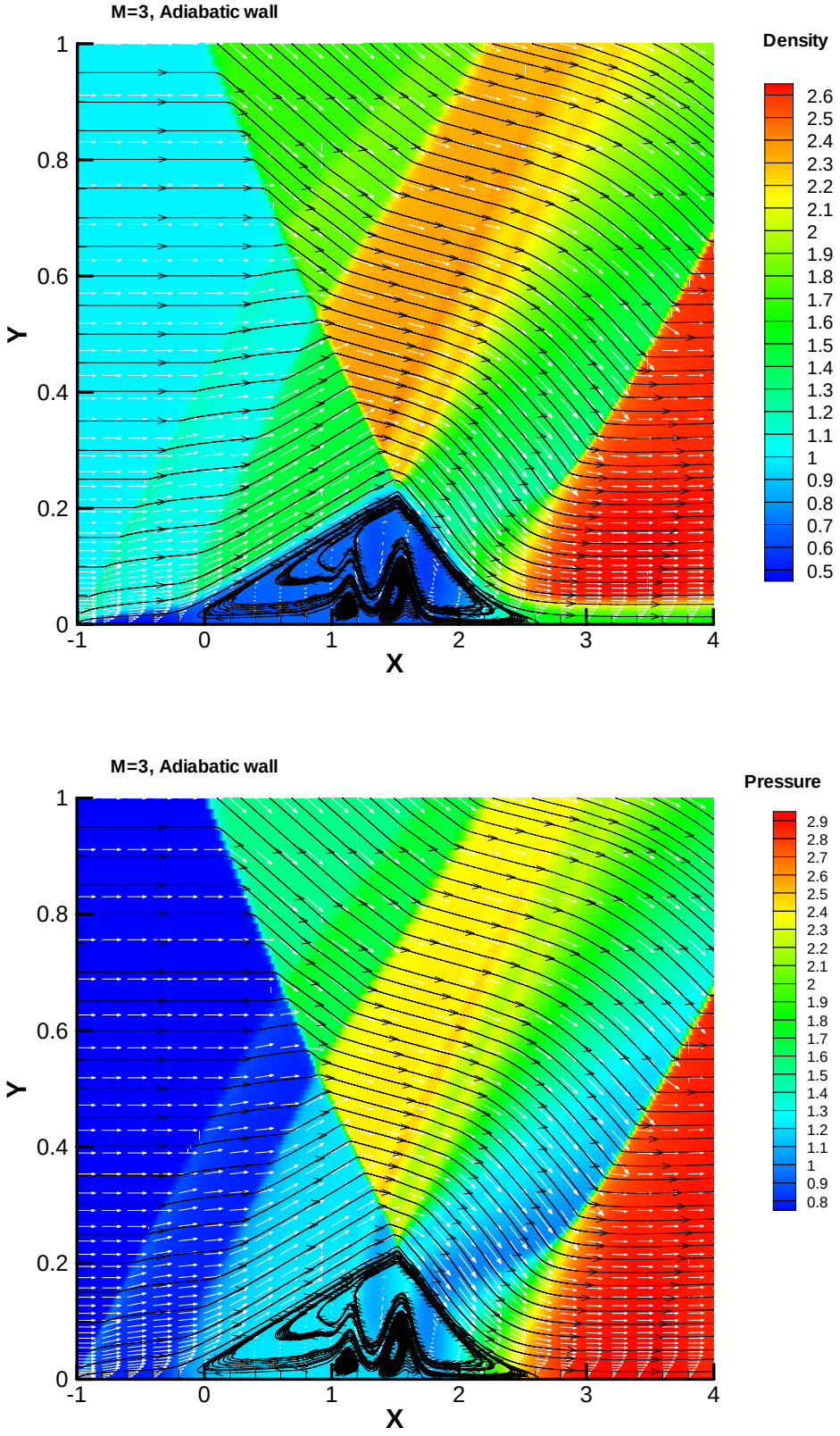


Figure 2. Flow field detail for shock boundary layer interaction at $M_\infty = 2.9$, $\theta = 29^o$ and laminar flow $Re_L = 10^4$ / unit length

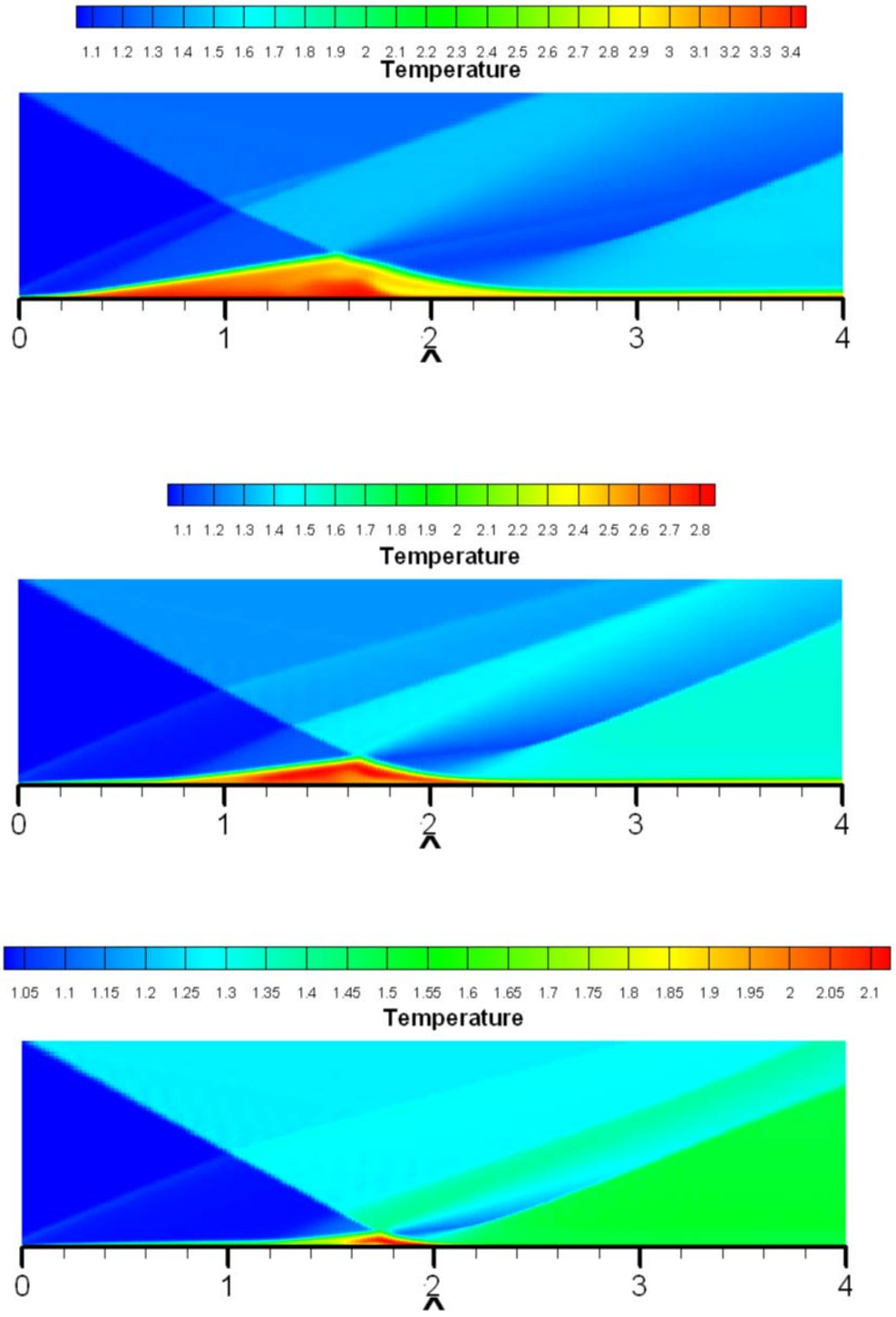


Figure 3. Effect of surface temperature on shock boundary layer interaction at $M_\infty = 2.9$, $\theta = 29^o$ and laminar flow $\mathrm{Re}_L = 10^4$ / unit ; top heated wall $T_w = 3.5T_\infty$; middle $T_w = 2.5T_\infty$ (approximately adiabatic); and bottom wall cooling $T_w = 1.5T_\infty$

Quarterly Progress Report for the
EOARD Project
Development of High–Order Methods for Multi–Physics Problems Governed by Hyperbolic Equations
by
John A. Ekaterinaris, Constantine Kontzialis

FORTH / IACM
Award No. FA 8655–03–1–3085
Reporting period Oct 31 – March 31, 2009

0.1 Governing aerodynamic equations

0.2 Euler equations

The motion of a fluid without any viscous and thermal conduction effects is described by the system of *Euler* equations. The differential form of this system is:

$$\mathcal{L}(\mathbf{U}) = \frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot F(\mathbf{U}) = 0, \tag{1}$$

where $\mathbf{U}$ is the *conservative* variable state vector:

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix},$$

and $F(\mathbf{U})$ is the inviscid flux tensor with vector components:

$$\mathbf{f} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (\rho E + p)u \end{pmatrix}, \quad \mathbf{g} = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (\rho E + p)v \end{pmatrix}.$$

The specific energy E is the sum of the specific internal energy e and the kinetic energy:

$$E = e + \frac{1}{2}(u^2 + v^2). \tag{2}$$

Furthermore, with the assumption of a calorically perfect gas the specific internal energy and pressure, are related through the constitutive relations:

$$e = C_V T, \qquad p = (\gamma - 1)\left[\rho E - \frac{\rho}{2}(u^2 + v^2)\right].$$

0.3 Discretization method

The governing equations of fluid motion, given in Eq. (1), are discretized with the *Discontinuous Galerkin* (DG) method.

0.4 Spatial Discontinuous Galerkin discretization

0.4.1 Discontinuous weak formulation for the *Euler* equations

A tessellation of an arbitrary domain Ω, of the flow field, into non-overlapping elements Ω_e is considered:

$$\Omega = \bigcup_e \Omega_e,$$

and defines the computational mesh. The tessellation may be composed of either straight sided or curvilinear general shaped elements. The physical field over domain Ω is approximated by a C^{-1} function $\tilde{\mathbf{U}}(\mathbf{x}, t)$, constructed by a locally continuous approximation $\tilde{\mathbf{U}}_e(\mathbf{x}, t)$ over each element Ω_e, as depicted in Fig. 1:

$$\tilde{\mathbf{U}}(\mathbf{x}, t) = \sum_e \tilde{\mathbf{U}}_e(\mathbf{x}, t).$$

Considering a single element Ω_e, the approximation $\tilde{\mathbf{U}}_e(\mathbf{x}, t)$ substituted in Eq. (1), results in an approximation error, or *residual*, of the field over the elemental region, equal to:

$$\mathcal{R}_e(\tilde{\mathbf{U}}_e(\mathbf{x}, t)) = \mathcal{L}(\tilde{\mathbf{U}}_e(\mathbf{x}, t)), \tag{3}$$

leading to the residual over domain Ω:

$$\mathcal{R}(\tilde{\mathbf{U}}(\mathbf{x}, t)) = \sum_e \mathcal{R}_e(\tilde{\mathbf{U}}_e(\mathbf{x}, t)). \tag{4}$$

As a finite element method, the DG discretization, minimizes the approximation error over the computational domain in the weighted residual sense. That is, forming the *Legendre* inner product of $\mathcal{R}(\tilde{\mathbf{U}}(\mathbf{x}, t))$ with an arbitrary test or weight function v_j, and setting it equal to zero:

$$(\mathcal{R}(\tilde{\mathbf{U}}(\mathbf{x}, t)), v_j) = 0,$$

leads to:

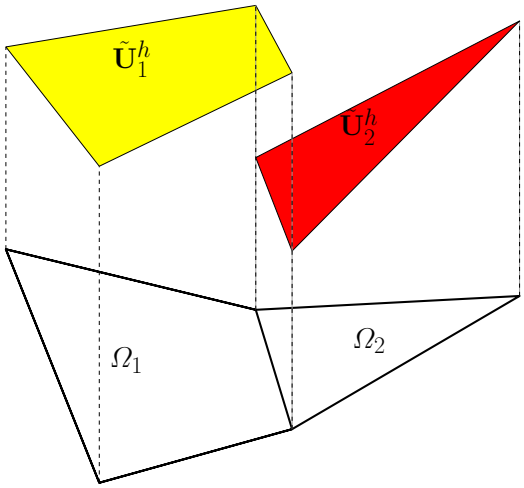


Figure 1: Local reconstruction of the field at two adjacent elements.

$$\int_\Omega \left[v_j \frac{\partial \tilde{\mathbf{U}}}{\partial t} + v_j \nabla \cdot F(\tilde{\mathbf{U}}) \right] d\Omega = 0 \Rightarrow$$

$$\int_\Omega v_j \frac{\partial \tilde{\mathbf{U}}}{\partial t} d\Omega - \int_\Omega \nabla v_j \cdot F(\tilde{\mathbf{U}}) d\Omega + \int_\Omega \nabla \cdot v_j F(\tilde{\mathbf{U}}) d\Omega = 0.$$

Using *Gauss'* theorem, the last equation results in:

$$\int_\Omega v_j \frac{\partial \tilde{\mathbf{U}}}{\partial t} d\Omega - \int_\Omega \nabla v_j \cdot F(\tilde{\mathbf{U}}) d\Omega + \oint_{\partial\Omega} v_j F(\tilde{\mathbf{U}}) \cdot \mathbf{n} \partial l = 0, \tag{5}$$

where $\mathbf{n}$ is the normalized outward vector at the domain boundary. Alternatively, Eq. (5) may be written as:

$$\sum_e \left[\int_{\Omega_e} v_j \frac{\partial \tilde{\mathbf{U}}_e}{\partial t} d\Omega_e - \int_{\Omega_e} \nabla v_j \cdot F(\tilde{\mathbf{U}}_e) d\Omega_e + \oint_{\partial\Omega_e} v_j F(\tilde{\mathbf{U}}_e) \cdot \mathbf{n_e} \partial l \right] = 0,$$

with $\mathbf{n_e}$ denoting the normalized outward vector at the element boundary. However, the discontinuous nature of the solution $\tilde{\mathbf{U}}$, permits to write the residual for every element as a separate equation:

$$\int_{\Omega_e} v_j \frac{\partial \tilde{\mathbf{U}}_e}{\partial t} d\Omega_e - \int_{\Omega_e} \nabla v_j \cdot F(\tilde{\mathbf{U}}_e) d\Omega_e + \oint_{\partial\Omega_e} v_j F(\tilde{\mathbf{U}}_e) \cdot \mathbf{n_e} \partial l = 0. \tag{6}$$

Eq. (6) represents the weak formulation for the system of *Euler* equations, under the assumption of a C^{-1} approximation of the flow field over domain Ω.

The local approximation of the field is constructed by a linear combination of elemental functions $b_k^e(\mathbf{x})$:

$$\tilde{\mathbf{U}}_e(\mathbf{x},t) = \sum_k \hat{\mathbf{U}}_e^k(t) b_k^e(\mathbf{x}), \tag{7}$$

where the elements of the coefficient vector $\hat{\mathbf{U}}_e^k(t)$ denote the degrees of freedom (DOF) of the numerical solution at every element.

Substituting $\tilde{\mathbf{U}}_e(\mathbf{x},t)$ in Eq. (6), results:

$$\int_{\Omega_e} v_j b_k^e \frac{\partial \hat{\mathbf{U}}_e^k}{\partial t} d\Omega_e - \int_{\Omega_e} \nabla v_j \cdot F(\tilde{\mathbf{U}}_e) d\Omega_e + \oint_{\partial\Omega_e} v_j F(\tilde{\mathbf{U}}_e) \cdot \mathbf{n_e} \partial l = 0, \tag{8}$$

which is a system of 4DOF $\times$ 4DOF equations. As a *Galerkin* method implies, the trial functions v_j are equal to the elemental functions $b_k^e(\mathbf{x})$.

0.4.2 Elemental operations

In order to handle arbitrary geometries, it is imperative to numerically evaluate all the integrals appearing in Eq. (8):

$$\mathcal{M}_{kj} = \int_{\Omega_e} b_k^e b_j^e d\Omega_e, \tag{9}$$

$$\mathcal{V} = \int_{\Omega_e} \nabla b_k^e \cdot F(\tilde{\mathbf{U}}_e) d\Omega_e, \tag{10}$$

$$\mathcal{S} = \oint_{\partial\Omega_e} b_k^e F(\tilde{\tilde{U}}_{\mathbf{e}}) \cdot \mathbf{n_e} \partial l, \tag{11}$$

where in place of the test function v_j the basis function b_j^e has been substituted. Application of a Gauss type numerical integration, requires the definition of the standard element region Ω_{st} for quadrilateral elements, depicted in Fig. 2:
The integrals given in Eqs. (9) and (10) over the standard element region are equal to:

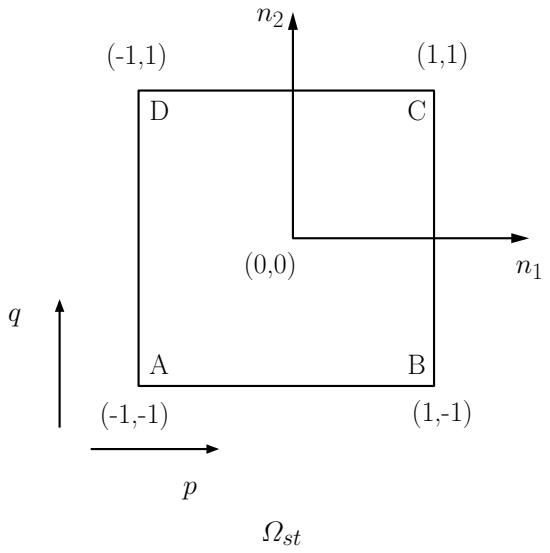


Figure 2: Standard quadrilateral element $n_1, n_2 \in [-1, 1]$.

$$\mathcal{M}_{kj} = \int_{-1}^{1}\int_{-1}^{1} b_k^e b_j^e |\mathbf{J}| dn_1 dn_2, \tag{12}$$

$$\mathcal{V} = \int_{-1}^{1}\int_{-1}^{1} \nabla_n b_k^e \cdot F(\hat{\mathbf{U}}_e^k)|\mathbf{J}| dn_1 dn_2, \tag{13}$$

where $\mathbf{J}$ is the *Jacobian* matrix of the transformation from the physical to the computational space. For straight sided elements, the mapping from the physical configuration of the element Ω_e to the standard region Ω_{st}, is given by the following relations for quadrilateral elements:

$$\begin{aligned} x =& x_A \frac{(1-n_1)}{2}\frac{(1-n_2)}{2} + x_B \frac{(1+n_1)}{2}\frac{(1-n_2)}{2} \\ &+ x_D \frac{(1-n_1)}{2}\frac{(1+n_2)}{2} + x_C \frac{(1+n_1)}{2}\frac{(1+n_2)}{2}, \end{aligned} \tag{14}$$

$$\begin{aligned} y =& y_A \frac{(1-n_1)}{2}\frac{(1-n_2)}{2} + y_B \frac{(1+n_1)}{2}\frac{(1-n_2)}{2} \\ &+ y_D \frac{(1-n_1)}{2}\frac{(1+n_2)}{2} + y_C \frac{(1+n_1)}{2}\frac{(1+n_2)}{2}, \end{aligned} \tag{15}$$

with its inverse:

$$\mathbf{J}^{-1} = \frac{1}{|\mathbf{J}|} \begin{bmatrix} \frac{\partial y}{\partial n_2} & -\frac{\partial x}{\partial n_1} \\ -\frac{\partial y}{\partial n_2} & \frac{\partial x}{\partial n_1} \end{bmatrix}.$$

0.4.3 Basis functions

The choice of the basis functions affects the behavior and computational efficiency of the DG discretization. For every element a finite element space $\mathcal{X}$ of functions is defined, as the set of orthogonal polynomials $\mathcal{P}_k$ in the interval $[-1, 1]$, up to an arbitrary order N:

$$\mathcal{X} = \{\mathcal{P}_k \mid (\mathcal{P}_k, \mathcal{P}_j) = C(k,j)\delta_{kj}\}, \quad , 0 \leq k, j \leq N, \quad C \in \Re.$$

For rectangular elements, the basis is constructed using the tensorial product of *Legendre* polynomials $\hat{\psi}_p(n)$ according to a specific indexing:

$$b_k = \hat{\psi}_p(n_1)\hat{\psi}_q(n_2), \quad -1 \leq n_1, n_2 \leq 1 \tag{16}$$

with

$$k = p + q(N+1), \qquad 0 \leq p, q \leq N.$$

If a local approximation of order N is applied, then the number of bases in $\mathcal{X}$ is equal to $(N+1)^2$.

In Table 1 the analytical expressions of the basis functions, over the standard quadrilateral region are given, and plotted in Fig. 3.

0.5 Applications

0.6 Isentropic vortex convection

In Fig. 4 it is shown the computational mesh for the advection of an isentropic vortex. The mesh consists of 2500 quadrilateral elements. A third order accurate (P2) scheme is used over a time interval of 2 time units.

Fig. 5 depicts the initial pressure profile and in Fig. 6 it is shown the pressure at the end of the simulation time.

p, q	b_k
$0, 0$	1
$1, 0$	n_1
$2, 0$	$\frac{3n_1+1}{2}$
$0, 1$	n_2
$1, 1$	$n_1 n_2$
$2, 1$	$\frac{3n_1+1}{2} n_2$
$0, 2$	$\frac{3n_2+1}{2}$
$1, 2$	$n_1 \frac{3n_2+1}{2}$
$2, 2$	$\frac{3n_1+1}{2} \frac{3n_2+1}{2}$

Table 1: Basis functions for a second order approximation over the standard quadrilateral region.

0.7 Flow around a cylinder

A cylinder with a diameter of 1 unit is considered. The mesh in its vicinity is shown in Fig. 7 and it consists of 3968 quadrilaterals.

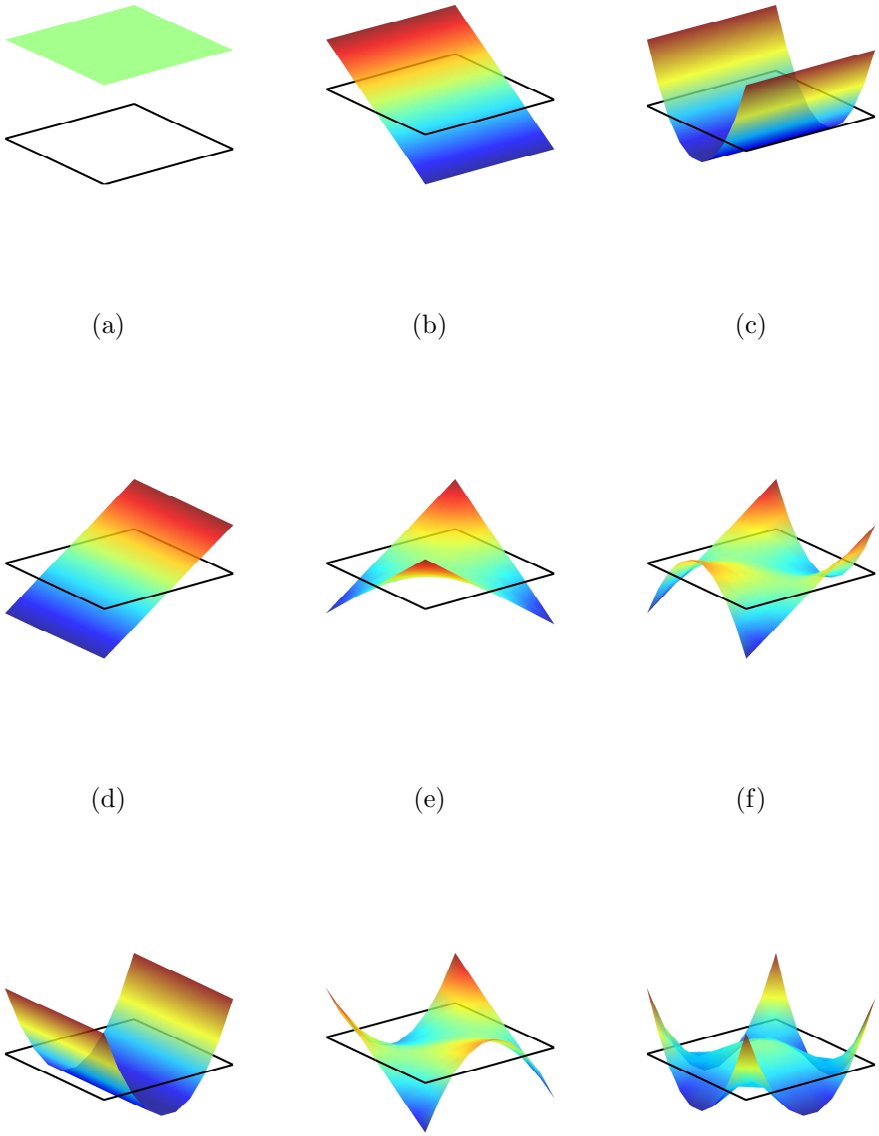


Figure 3: Basis functions over the standard quadrilateral region: (a) $(p, q) = (0, 0)$, (b) $(p, q) = (1, 0)$, (c) $(p, q) = (2, 0)$, (d) $(p, q) = (0, 1)$, (e) $(p, q) = (1, 1)$, (f) $(p, q) = (2, 1)$, (g) $(p, q) = (0, 2)$, (h) $(p, q) = (1, 2)$, (i) $(p, q) = (2, 2)$.

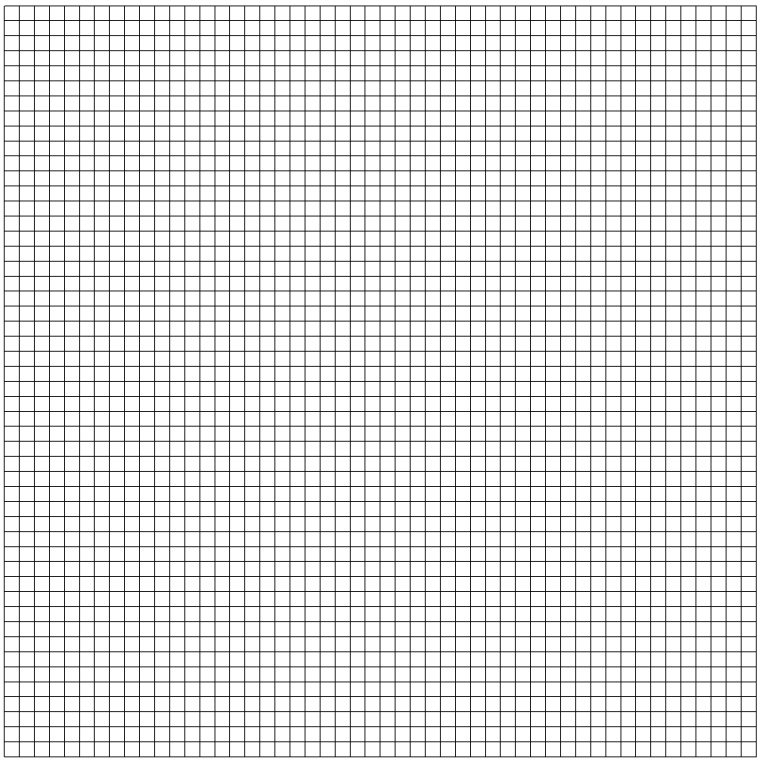


Figure 4: Computational domain for isentropic vortex convection.

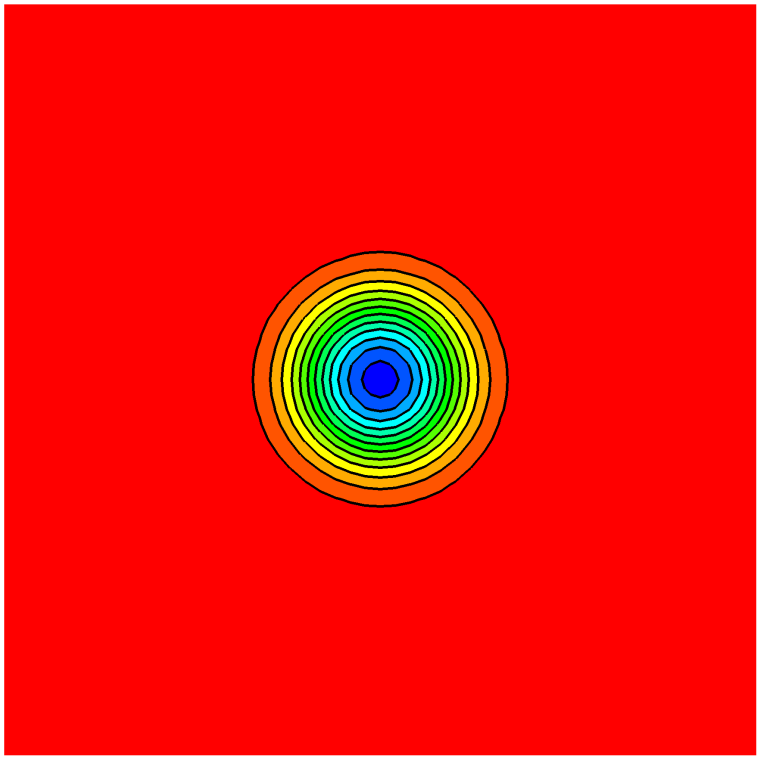


Figure 5: Initial pressure.

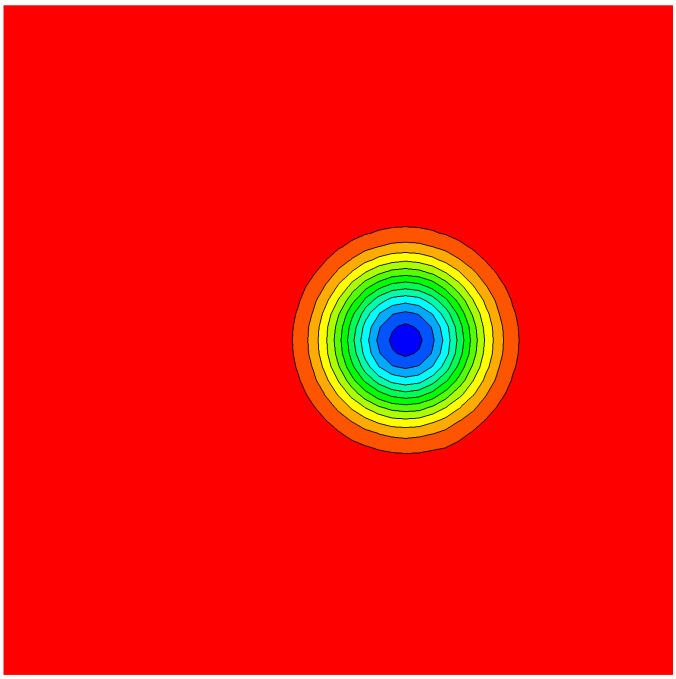


Figure 6: Final pressure (P2)).

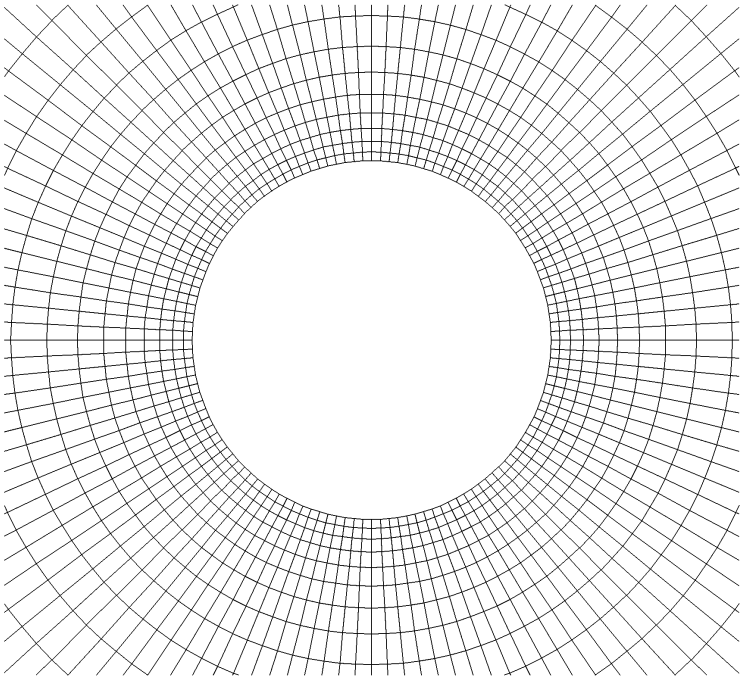


Figure 7: Mesh around a cylinder.

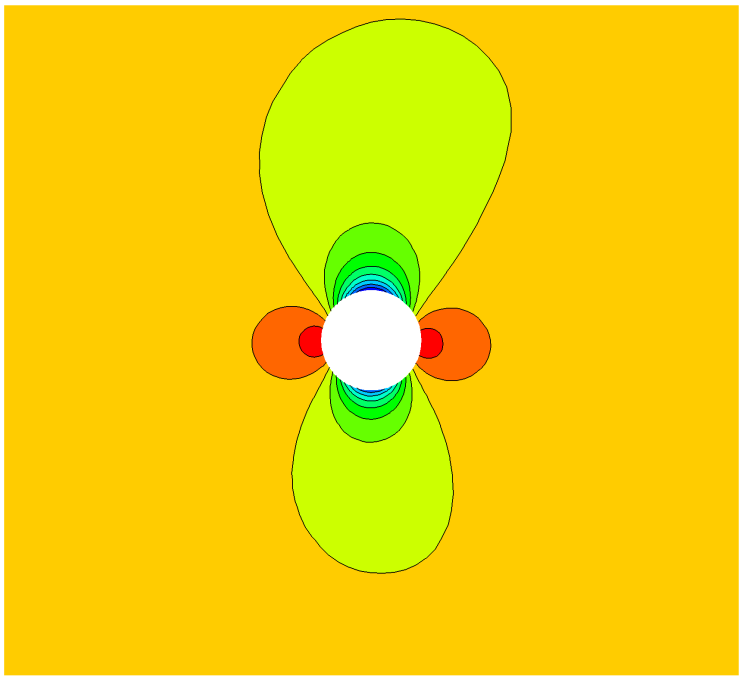


Figure 8: Pressure contours around cylinder.

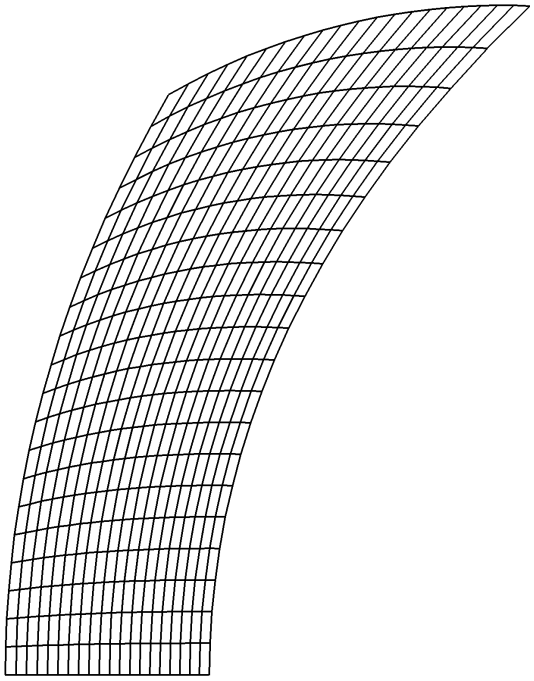


Figure 9: Mesh for ringleb flow.

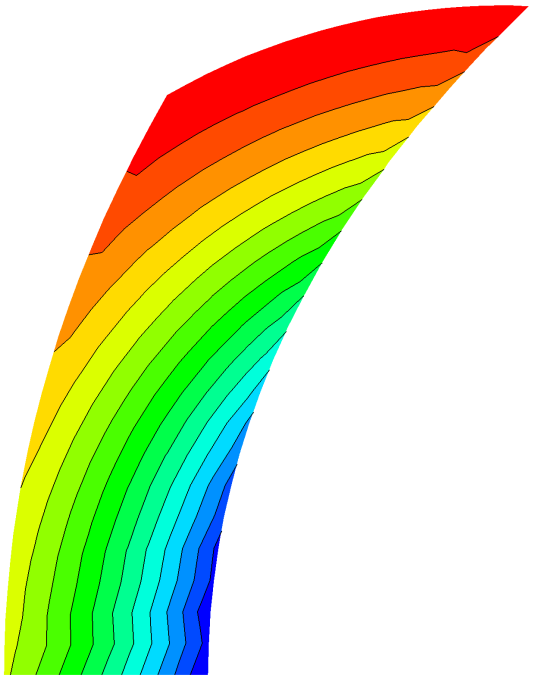


Figure 10: Density contours for ringleb flow (P1).

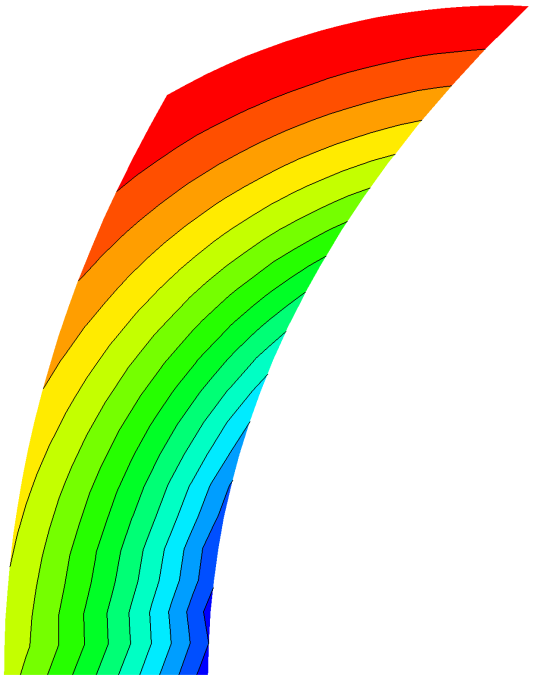


Figure 11: Pressure contours for ringleb flow (P1).

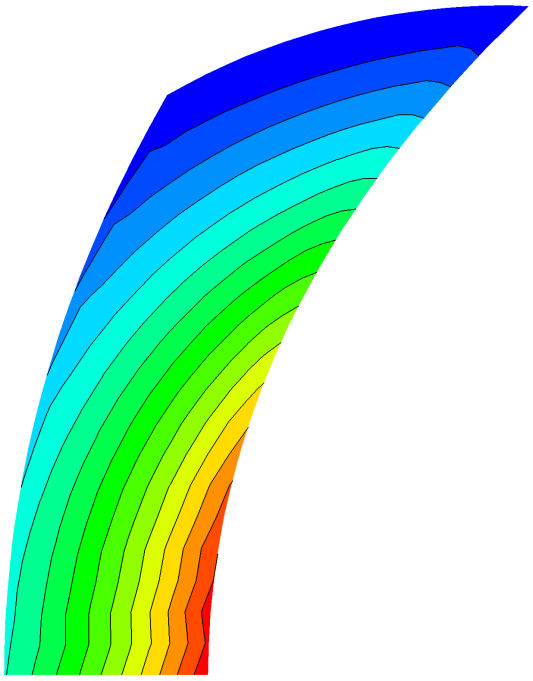


Figure 12: Mach number contours for ringleb flow (P1).

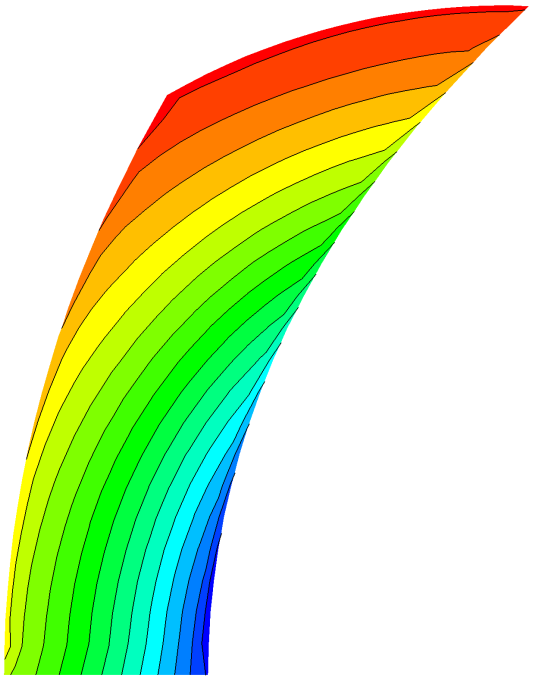


Figure 13: Density contours for ringleb flow (P2).

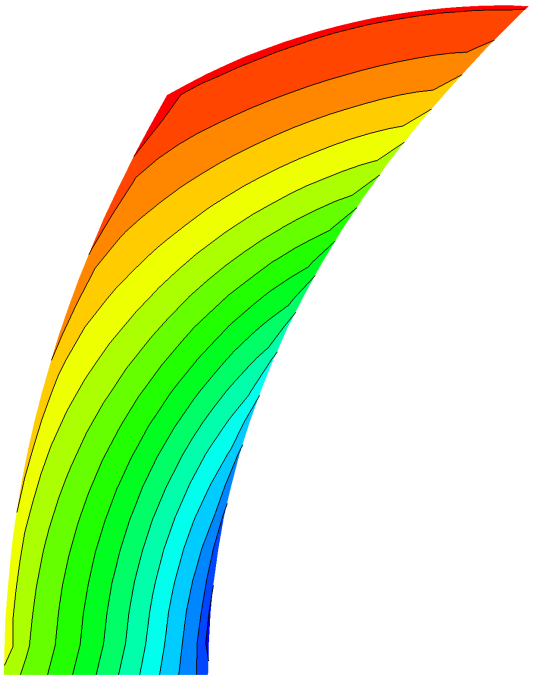


Figure 14: Pressure contours for ringleb flow (P2).

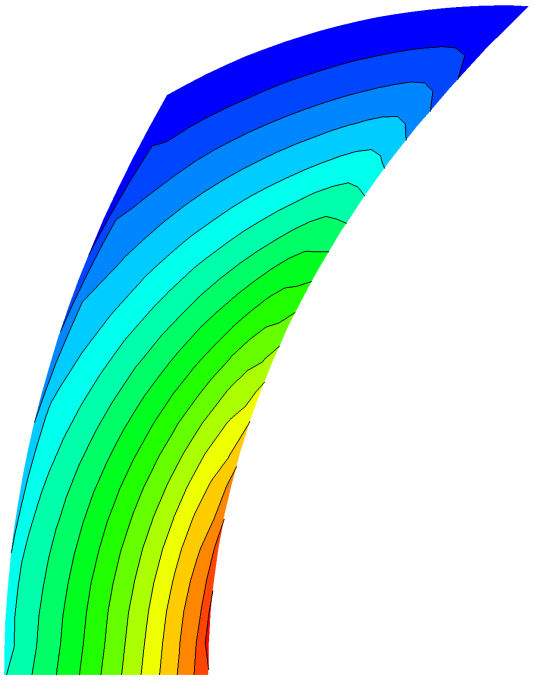


Figure 15: Mach number contours for ringleb flow (P2).

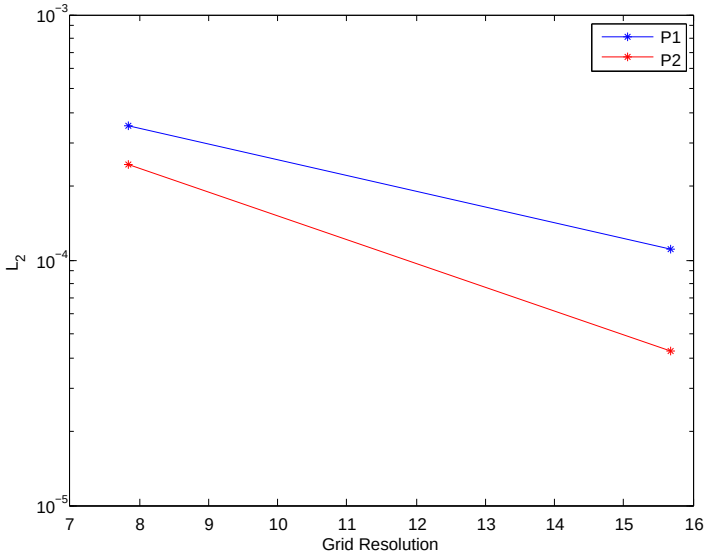


Figure 16: Convergence rates for ringleb flow.

Quarterly Progress Report for the
EOARD Project

Development of High–Order Methods for Multi–Physics Problems Governed by Hyperbolic Equations

by

John A. Ekaterinaris

FORTH / IACM

Award No. FA 8655–03–1–3085

Reporting period March 1 – May30, 2009

Discretization method

The governing equations of fluid motion are discretized with the *Discontinuous Galerkin* (DG) method. A remarkable advantage of this method is that the local support of the solution at the element level, makes it suitable for flow simulations with discontinuities with high-order of accuracy, and moreover, mesh distortion does not affect the quality of the solution considerably as the order of the scheme increases. Furthermore, p-adaptation is easy to implement permitting the increase resolution of the flow field at certain regions.

1 Spatial Discontinuous Galerkin discretization

1.1 Discontinuous weak formulation for the *Euler* equations

A tessellation of an arbitrary domain Ω, of the flow field, into elements Ω_e is considered:

$$\Omega = \bigcup_e \Omega_e,$$

and defines the computational mesh. The tessellation may be composed of either straight sided or curvilinear general shaped elements. The physical field over domain Ω is approximated by a C^{-1} function $\tilde{\mathbf{U}}(\mathbf{x}, t)$, constructed by a locally continuous approximation $\tilde{\mathbf{U}}_e(\mathbf{x}, t)$ over each element Ω_e, as depicted in Fig. 1:

$$\tilde{\mathbf{U}}(\mathbf{x}, t) = \sum_e \tilde{\mathbf{U}}_e(\mathbf{x}, t).$$

Considering a single element Ω_e, the approximation $\tilde{\mathbf{U}}_e(\mathbf{x}, t)$ substituted in the system of *Euler* equations results in an approximation error, or *residual*, of the field over the elemental region, equal to:

$$\mathcal{R}_e(\tilde{\mathbf{U}}_e(\mathbf{x}, t)) = \mathcal{L}(\tilde{\mathbf{U}}_e(\mathbf{x}, t)), \tag{1}$$

leading to the residual over domain Ω:

$$\mathcal{R}(\tilde{\mathbf{U}}(\mathbf{x}, t)) = \sum_e \mathcal{R}_e(\tilde{\mathbf{U}}_e(\mathbf{x}, t)). \tag{2}$$

As a finite element method, the DG discretization, minimizes the approximation error over the computational domain in the weighted residual sense. That is, forming the *Legendre* inner product of $\mathcal{R}(\tilde{\mathbf{U}}(\mathbf{x}, t))$ with an arbitrary test or weight function v, and setting it equal to zero:

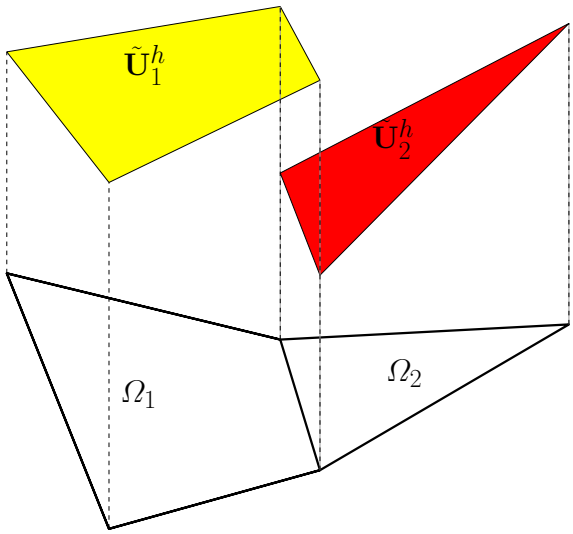


Figure 1: Local reconstruction of the field at two adjacent elements.

$$(\mathcal{R}(\tilde{\mathbf{U}}(\mathbf{x},t)),v)=0,$$

leads to:

$$\int_\Omega \left[v\frac{\partial \tilde{\mathbf{U}}}{\partial t} + v\nabla\cdot F(\tilde{\mathbf{U}}) \right] d\Omega = 0 \Rightarrow$$

$$\int_\Omega v\frac{\partial \tilde{\mathbf{U}}}{\partial t}d\Omega - \int_\Omega \nabla v\cdot F(\tilde{\mathbf{U}})d\Omega + \int_\Omega \nabla\cdot vF(\tilde{\mathbf{U}})d\Omega = 0.$$

Using *Gauss'* theorem, the last equation results in:

$$\int_\Omega v\frac{\partial \tilde{\mathbf{U}}}{\partial t}d\Omega - \int_\Omega \nabla v\cdot F(\tilde{\mathbf{U}})d\Omega + \oint_{\partial\Omega} vF(\tilde{\mathbf{U}})\cdot\mathbf{n}\partial l = 0, \tag{3}$$

where $\mathbf{n}$ is the normalized outward vector at the domain boundary. Alternatively, Eq. (3) may be written as:

$$\sum_e \left[\int_{\Omega_e} v\frac{\partial \tilde{\mathbf{U}}_e}{\partial t}d\Omega_e - \int_{\Omega_e} \nabla v\cdot F(\tilde{\mathbf{U}}_e)d\Omega_e + \oint_{\partial\Omega_e} vF(\tilde{\mathbf{U}}_e)\cdot\mathbf{n_e}\partial l \right] = 0,$$

with $\mathbf{n_e}$ denoting the normalized outward vector at the element boundary. However, the discontinuous nature of the solution $\tilde{\mathbf{U}}$, permits to write the residual for every element as a separate equation:

$$\int_{\Omega_e} v\frac{\partial \tilde{\mathbf{U}}_e}{\partial t}d\Omega_e - \int_{\Omega_e} \nabla v \cdot F(\tilde{\mathbf{U}}_e)d\Omega_e + \oint_{\partial\Omega_e} vF(\tilde{\mathbf{U}}_e)\cdot \mathbf{n_e}\partial l = 0. \tag{4}$$

Eq. (4) represents the weak formulation for the system of *Euler* equations, under the assumption of a C^{-1} approximation of the flow field over domain Ω.

The local approximation of the field is constructed by a linear combination of elemental basis functions $b_k^e(\mathbf{x})$:

$$\tilde{\mathbf{U}}_e(\mathbf{x},t) = \sum_k \hat{\mathbf{U}}_e^k(t) b_k^e(\mathbf{x}), \tag{5}$$

where the coefficient vector $\hat{\mathbf{U}}_e^k(t)$ denotes the degrees of freedom (DOF) of the numerical solution at every element.

Substituting $\tilde{\mathbf{U}}_e(\mathbf{x},t)$ in Eq. (4), results:

$$\int_{\Omega_e} vb_k^e\frac{\partial \hat{\mathbf{U}}_e^k}{\partial t}d\Omega_e - \int_{\Omega_e} \nabla v \cdot F(\tilde{\mathbf{U}}_e)d\Omega_e + \oint_{\partial\Omega_e} vF(\tilde{\mathbf{U}}_e)\cdot \mathbf{n_e}\partial l = 0, \tag{6}$$

As a *Galerkin* method implies, the trial function v is equal to the elemental function $b_k^e(\mathbf{x})$, resulting in a system of 4DOF $\times$ 4DOF equations for the system of *Euler* equations in two dimensions.

1.2 Basis functions and standard elemental configuration

The choice of the basis functions affects the behavior and computational efficiency of the DG discretization [2–4]. When constructing a p-type basis it is advantageous to use the orthogonal set of polynomials called Jacobi polynomials. Three types of basis functions may be constructed: *modal*, *nodal* and *mixed modal-nodal*.

The basis functions are defined over a standard elemental configuration, which permits their systematic construction and implementation of any numerical operation involving them.

Application of a Gauss type numerical integration, requires the definition of a standard elemental region spanning the domain $[-1,1]\times[-1,1]$. In Fig. 2, the transformation of the physical quadrilateral to its standard elemental configuration Ω_{st}^q is depicted, and in Fig. 3, the transformation of the physical triangle to its standard elemental configuration Ω_{st}^t is depicted, along with the Ω_{st}^q region, where the basis functions are to be constructed and all the numerical operations are performed.

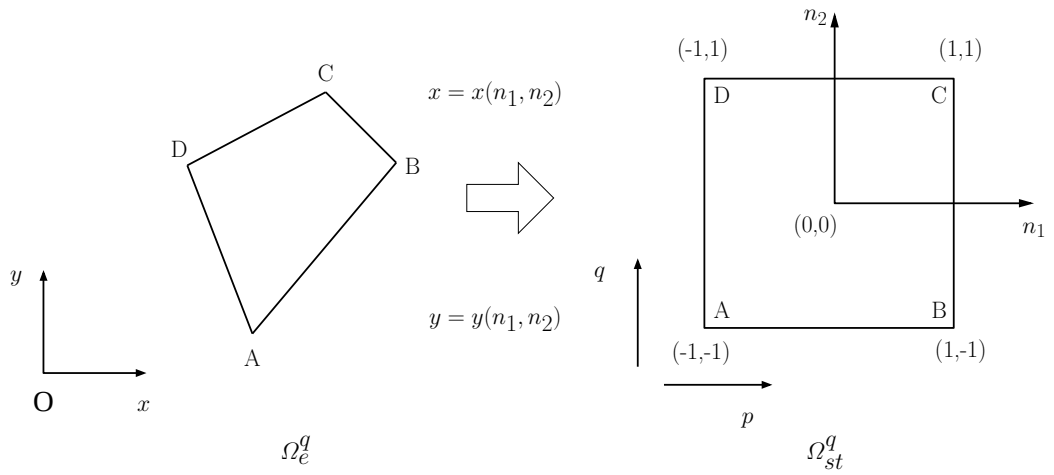


Figure 2: Transformation of the physical quadrilateral to the standard quadrilateral element configuration ($n_1, n_2 \in [-1, 1]$).

For every element a finite element space $\mathcal{X}$ of modal functions is defined, as the set of orthogonal polynomials $\mathcal{P}_k$ in the interval $[-1, 1]$, up to an arbitrary order N:

$$\mathcal{X} = \{\mathcal{P}_k \mid (\mathcal{P}_k, \mathcal{P}_j) = C(k, j)\delta_{kj}\}, \quad 0 \leq k, j \leq N, \quad C \in \Re.$$

For quadrilateral elements, the basis is constructed using the tensorial product of *Legendre* polynomials $\hat{\psi}_p(n)$ according to a specific indexing, over Ω_{st}^q [2]:

$$b_k(n_1, n_2) = \hat{\psi}_p(n_1)\hat{\psi}_q(n_2), \quad -1 \leq n_1, n_2 \leq 1 \tag{7}$$

with:

$$k = p + q(N + 1), \qquad 0 \leq p, q \leq N.$$

If a local approximation of order N is applied, then the number of bases in $\mathcal{X}$ is equal to $(N + 1)^2$.

For triangular elements, the basis is defined over Ω_{st}^t and constructed using the tensorial product of *Jacobi* polynomials over the region Ω_{st}^q (Fig. 3) [2]:

$$b_k(\xi_1, \xi_2) = P_p^{0,0}(n_1)(\frac{1 - n_2}{2})^p P_q^{2p+1,0}(n_2), \tag{8}$$

with:

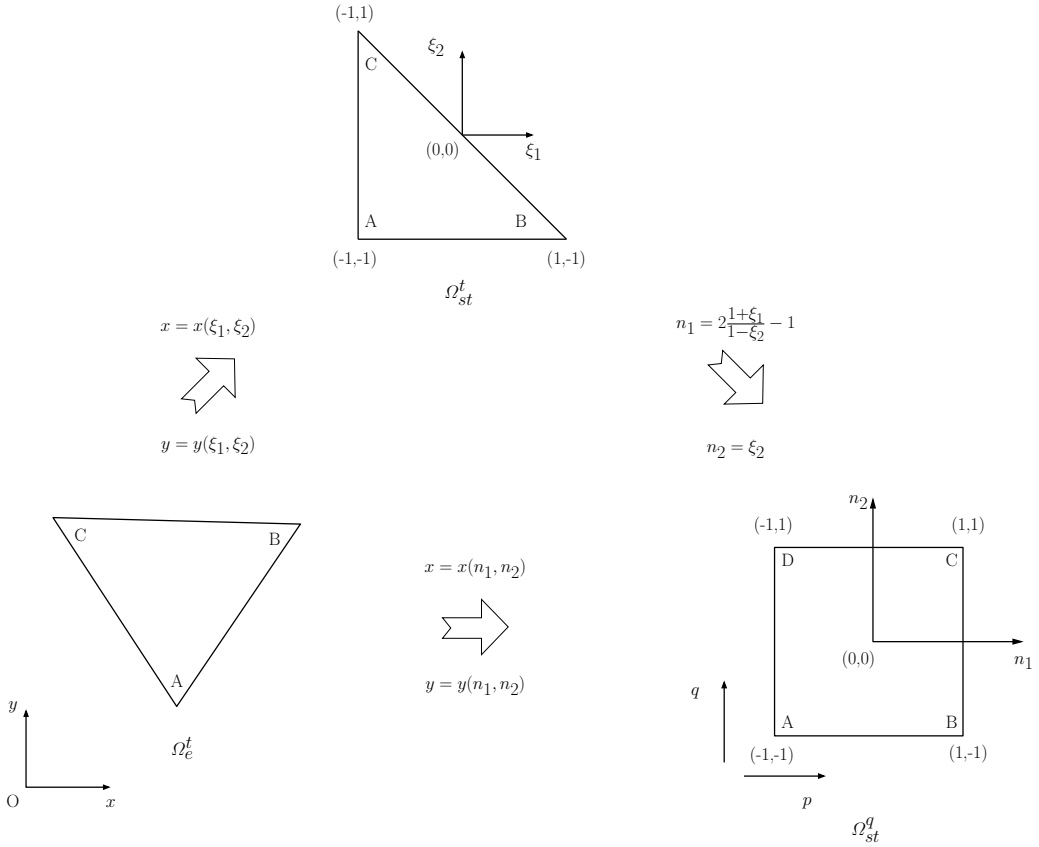


Figure 3: Transformation of the physical triangle to the standard triangular element ($\xi_1, \xi_2 \in [-1, 1]$ with $\xi_1 + \xi_2 \leq 1$) and standard quadrilateral element configuration ($n_1, n_2 \in [-1, 1]$).

$$\xi_1 = \frac{(1+n_1)(1-n_2)}{2} - 1, \quad \xi_2 = n_2,$$

and according to the following indexing [5]:

$$k = p + (N+1)q - q\frac{q-1}{2}, \quad 0 \leq p, q \leq N \text{ with } p + q \leq N.$$

For a triangular element the number of basis for a local approximation of order N, is equal to $\frac{(N+1)(N+2)}{2}$.

In Tables 1 and 2 the analytical expressions of the basis functions, over the standard quadrilateral and triangular region, for a third order expansion are given, and plotted in Figs. 4 and 5, respectively.

p,q	b_k	p,q	b_k	p,q	b_k
$(0,0)$	1	$(1,0)$	n_1	$(2,0)$	$\frac{3n_1+1}{2}$
$(0,1)$	n_2	$(1,1)$	$n_1 n_2$	$(2,1)$	$\frac{3n_1+1}{2} n_2$
$(0,2)$	$\frac{3n_2+1}{2}$	$(1,2)$	$n_1 \frac{3n_2+1}{2}$	$(2,2)$	$\frac{(3n_1+1)}{2} \frac{(3n_2+1)}{2}$

Table 1: Basis functions for a third order approximation over the standard quadrilateral region.

p,q	b_k	p,q	b_k	p,q	b_k
$(0,0)$	1	$(1,0)$	$n_1 \frac{1-n_2}{2}$	$(2,0)$	$\frac{3}{2}(n_1-1)^2 + 3n_1 - 2(\frac{1-n_2}{2})^2$
$(0,1)$	$\frac{3n_2+1}{2}$	$(1,1)$	$n_1 \frac{1-n_2}{2} \frac{5n_2+3}{2}$	$(0,2)$	$\frac{5}{2}(n_2-1)^2 + 6n_2 - 3$

Table 2: Basis functions for a third order approximation over the standard triangular region.

1.3 Elemental operations

In order to handle arbitrary geometries, it is imperative to numerically evaluate all the integrals appearing in Eq. (6):

$$\mathcal{M}_{kj} = \int_{\Omega_e} b_k^e b_j^e d\Omega_e, \tag{9}$$

$$\mathcal{V} = \int_{\Omega_e} \nabla b_k^e \cdot F(\tilde{\mathbf{U}}_e) d\Omega_e, \tag{10}$$

$$\mathcal{S} = \oint_{\partial\Omega_e} b_k^e F(\tilde{\mathbf{U}}_e) \cdot \mathbf{n_e} \partial l, \tag{11}$$

where in place of the test function v the elemental basis function b_j^e has been substituted. In the present work Gaussian numerical integration is used.

For quadrilateral elements, the integrals given in Eqs. (9) and (10) are then computed by the weighted summations ($\mathbf{n} = n_{1,i}, n_{2,m}$):

$$\begin{aligned} \mathcal{M}_{kj} = \int_{-1}^{1}\int_{-1}^{1} b_k b_j |\mathbf{J}| dn_1 dn_2 = \\ \sum_{i=1}^{Q_1}\sum_{m=1}^{Q_2} w_i w_m b_k(\mathbf{n}) b_j(\mathbf{n}) |\mathbf{J}(\mathbf{n})|, \end{aligned} \tag{12}$$

$$\begin{aligned}\mathcal{V} = &\int_{-1}^{1}\int_{-1}^{1} \mathbf{J^{-1}} \cdot \nabla b_k \cdot F(\tilde{\mathbf{U}}_e)|\mathbf{J}|dn_1dn_2 = \\ &\sum_{i=1}^{Q_1}\sum_{m=1}^{Q_2} w_i w_m \mathbf{J^{-1}}(\mathbf{n}) \cdot \nabla b_k(\mathbf{n}) \cdot F(\tilde{\mathbf{U}}_e(\mathbf{n})|\mathbf{J}(\mathbf{n})|,\end{aligned} \tag{13}$$

and for triangular elements, the same integrals are computed by the following weighted summations:

$$\begin{aligned}\mathcal{M}_{kj} = &\int_{-1}^{1}\int_{-1}^{-\xi_2} b_k b_j |\mathbf{J}| d\xi_1 d\xi_2 = \\ &\int_{-1}^{1}\int_{-1}^{1} b_k b_j |\mathbf{J}| (\frac{1-n_2}{2}) dn_1 dn_2 = \\ &\sum_{i=1}^{Q_1}\sum_{m=1}^{Q_2} w_i w_m b_k(\mathbf{n}) b_j(\mathbf{n}) |\mathbf{J}(\mathbf{n})| \left(\frac{1-n_{2,m}}{2}\right),\end{aligned} \tag{14}$$

$$\begin{aligned}\mathcal{V} = &\int_{-1}^{1}\int_{-1}^{-\xi_2} \mathbf{J^{-1}} \cdot \nabla b_k \cdot F(\tilde{\mathbf{U}}_e)|\mathbf{J}| d\xi_1 d\xi_2 = \\ &\int_{-1}^{1}\int_{-1}^{1} \mathbf{J^{-1}} \cdot \nabla b_k \cdot F(\tilde{\mathbf{U}}_e)|\mathbf{J}| (\frac{1-n_2}{2}) dn_1 dn_2 \\ &\sum_{i=1}^{Q_1}\sum_{m=1}^{Q_2} w_i w_m \mathbf{J^{-1}}(\mathbf{n}) \cdot \nabla b_k(\mathbf{n}) \cdot F(\tilde{\mathbf{U}}_e(\mathbf{n}))|\mathbf{J}(\mathbf{n})| \left(\frac{1-n_{2,m}}{2}\right),\end{aligned} \tag{15}$$

where $|\mathbf{J}|$ is the *Jacobian* of the transformation, $\mathbf{J^{-1}}$ is the inverse *Jacobian* matrix of the transformation from the physical to the computational space and Q_1 and Q_2 is the number of quadrature points in the n_1 and n_2 direction respectively. Matrix $\mathcal{M}$ is called the mass matrix and possesses a diagonal structure only for straight sided triangles and for straight sided quadrilaterals with parallel edges.

The surface integral in Eq. (11) is computed by the following weighted summation, both for triangles and quadrilaterals:

$$\begin{aligned}\mathcal{S} = &\int_{-1}^{1} b_k F(\tilde{\mathbf{U}}_e) \cdot \mathbf{n_e} |l_e| dn = \\ &\sum_{m=1}^{Q_m} w_m b_k H(\tilde{\mathbf{U}}_e(n_m)) \cdot \mathbf{n_e^c} |l_e|,\end{aligned} \tag{16}$$

where Q_m is the number of quadrature points, $|l_e|$ is the edge length, H the numerical flux at the element boundary and $\mathbf{n_e^c}$ is the normalized outward vector of the local element edge at the computational space.

1.4 Numerical flux

The numerical flux is chosen as the local Lax-Friedrichs flux due to its low computational cost:

$$H(\tilde{\mathbf{U}}_e(n_m)) \cdot \mathbf{n_e^c} = \frac{1}{2}\left[(f^+ + f^-)n_x + (g^+ + g^-)n_y\right] - \frac{1}{2}k\left(\mathbf{U}_e^- - \mathbf{U}_e^+\right), \tag{17}$$

where the superscripts (+) and (-) denote the elements sharing the same edge. k is the spectral radius of the system in the direction normal to the edge and is equal to:

$$k = |u_n| + c,$$

and it is computed using the Roe averaged variables:

$$\tilde{\rho u} = \frac{\sqrt{\rho^+}u^+ + \sqrt{\rho^-}u^-}{\sqrt{\rho^+} + \sqrt{\rho^-}}, \tag{18}$$

$$\tilde{\rho v} = \frac{\sqrt{\rho^+}v^+ + \sqrt{\rho^-}v^-}{\sqrt{\rho^+} + \sqrt{\rho^-}}, \tag{19}$$

$$\tilde{h} = \frac{\sqrt{\rho^+}h^+ + \sqrt{\rho^-}h^-}{\sqrt{\rho^+} + \sqrt{\rho^-}}, \tag{20}$$

$$\tilde{c} = (\gamma - 1)\sqrt{\tilde{h} - \frac{1}{2}(\tilde{\rho u}^2 + \tilde{\rho v}^2)}, \tag{21}$$

with $h = E + p$.

1.5 Selection of Gauss type numerical integration

Gauss numerical integration is classified according to the way the end points of the domain of integration are handled [2]. The classification is given in Table 3, with the corresponding order of accuracy using Q quadrature points:

The GL type is used both for the volume and the surface integrations due to its minimum number of quadrature points needed for a specific order of accuracy. Moreover, in Eqs. (12), (14) and (16), it is seen that the product of two polynomials of order N is integrated. Thus, the same number of quadrature points is used for volume and surface integrations and equal to:

$$Q_1 = Q_2 = Q_m = N + 1 \tag{22}$$

Type	End points	Order of accuracy
Gauss-Legendre (GL)	no end points	$2Q-1$
Gauss-Radau (GR)	only one end point	$2Q-2$
Gauss-Lobatto-Legendre (GLL)	both end points	$2Q-3$

Table 3: Types of Gauss numerical integration.

1.6 Mapping between physical and computational space

The computation of the integrals in Eqs. (12), (14), (13) and (15) requires the evaluation of the *Jacobian* matrix at every quadrature point. This necessitates the construction of a mapping from the physical to the computational space. Usually, according to the polynomial degree N_B of the physical space representation, curved elements are classified in sub-, iso- and super-parametric elements, by comparing the order N_B with that of the order N of the discretization scheme. In detail:

$$\begin{aligned} N_B < N &\quad : \quad \text{sub-parametric element} \\ N_B = N &\quad : \quad \text{iso-parametric element} \\ N_B > N &\quad : \quad \text{super-parametric element} \end{aligned}$$

For straight sided elements, the mapping from the physical configuration of the element Ω_e to the standard elemental region, is given by the following relations for quadrilateral elements:

$$\begin{aligned} x = & x_A \frac{(1-n_1)}{2}\frac{(1-n_2)}{2} + x_B \frac{(1+n_1)}{2}\frac{(1-n_2)}{2} \\ & + x_D \frac{(1-n_1)}{2}\frac{(1+n_2)}{2} + x_C \frac{(1+n_1)}{2}\frac{(1+n_2)}{2}, \end{aligned} \tag{23}$$

$$\begin{aligned} y = & y_A \frac{(1-n_1)}{2}\frac{(1-n_2)}{2} + y_B \frac{(1+n_1)}{2}\frac{(1-n_2)}{2} \\ & + y_D \frac{(1-n_1)}{2}\frac{(1+n_2)}{2} + y_C \frac{(1+n_1)}{2}\frac{(1+n_2)}{2}, \end{aligned} \tag{24}$$

and for triangular elements:

$$x = x_A \frac{(1-n_1)}{2}\frac{(1-n_2)}{2} + x_B \frac{(1+n_1)}{2}\frac{(1-n_2)}{2} + x_C \frac{1+n_2}{2}, \tag{25}$$

$$y = y_A \frac{(1-n_1)}{2}\frac{(1-n_2)}{2} + y_B \frac{(1+n_1)}{2}\frac{(1-n_2)}{2} + y_C \frac{1+n_2}{2}, \tag{26}$$

The *Jacobian* matrix is defined as:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial n_1} & \frac{\partial x}{\partial n_2} \\ \\ \frac{\partial y}{\partial n_2} & \frac{\partial y}{\partial n_2} \end{bmatrix}, \tag{27}$$

with its inverse equal to:

$$\mathbf{J}^{-1} = \frac{1}{|\mathbf{J}|} \begin{bmatrix} \frac{\partial y}{\partial n_2} & -\frac{\partial x}{\partial n_2} \\ \\ -\frac{\partial y}{\partial n_1} & \frac{\partial x}{\partial n_1} \end{bmatrix}. \tag{28}$$

Numerical results

The numerical method described in the previous section is validated for the computation of the flow over a cylinder at a low Mach number ($M_\infty = 0.15$) in order to approach the incompressible flow limit. Solutions are computed with meshes of quatrilateral and triangular element. Implementation of mixed type meshes is underway.

2 Inviscid flow over a cylinder

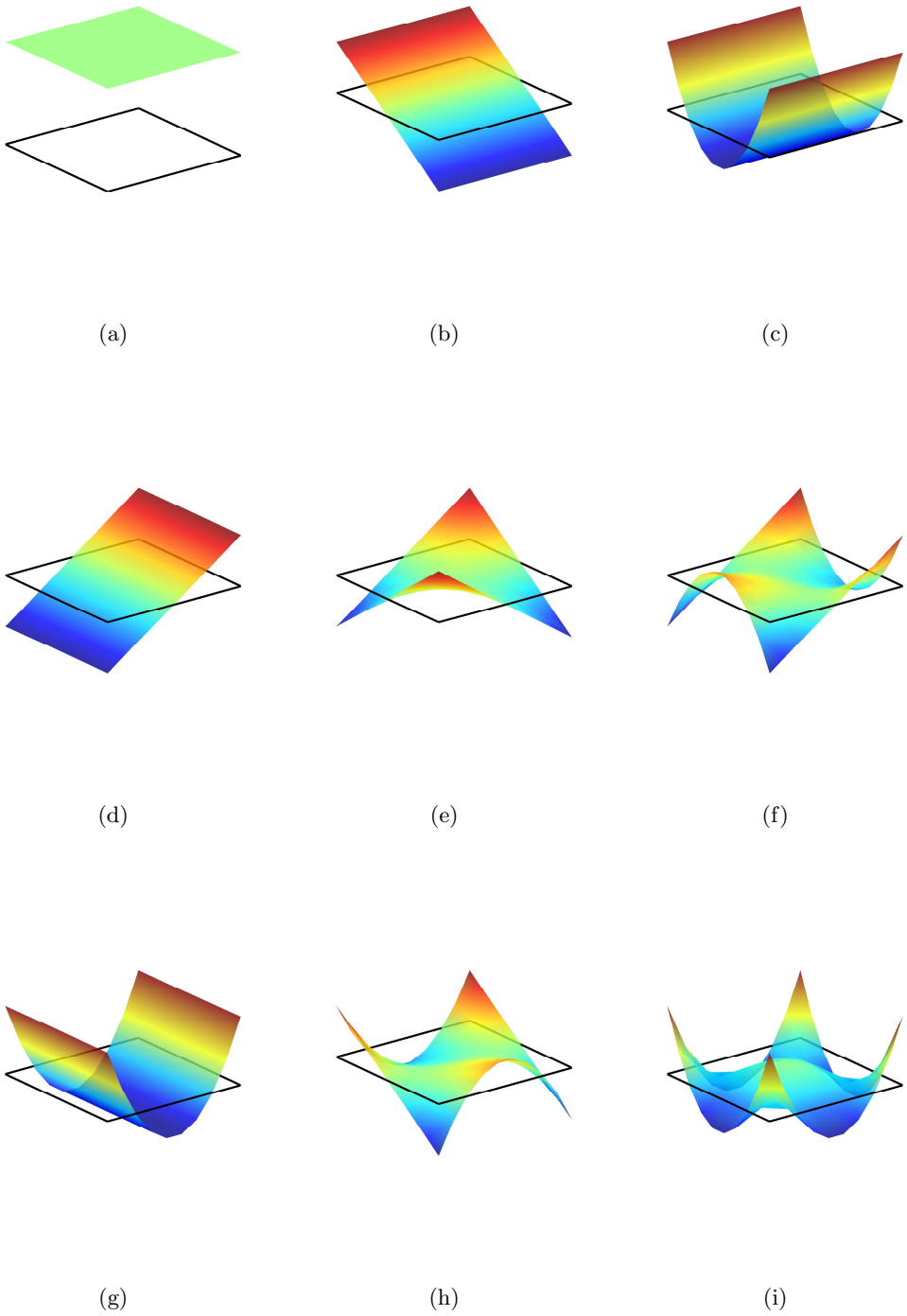
The analytic solution for inviscid, incompressible flow over a circular cylinder is the well known potential flow solution which is symmetric with respect to the y axis. The surface pressure coefficient of the potential flow solution is given by the following analytic expression:

$$C_p = 1 - 4\sin^2(\theta). \tag{29}$$

The quadrilateral element mesh used for the numerical solution is shown in Fig. 6. The computed Mach number with P1 polynomial basis is shown in Fig. 7. It is clear that an entropy layer is created and causes ficticious separation at the back stagnation point. This entropy layer can be significantly reduced and almost eliminated when the curved-element-type boundary condition (CBC) of [6] is used. The improvements achieved with the application of CBC improved boundary treatment is shown in Fig. 8. A quantititave comparson is shown in Fig. 9. Clearly the improvements of the CBC treatment are significant.

The triangular element mesh used for the numerical solution is shown in Fig. 10. The computed Mach number with P1 polynomial basis is shown in Fig. 11. It is clear again that an entropy layer is created. The increase of the order of accuracy does not diminish

the numerical entropy layer and the solution of Fig. 12 computed with P2 polynomial bases shows the same trend. The improvements achieved with the application of CBC improved boundary treatment is shown in Fig. 13. A quantititave comparsons for the solutions computed with P1 and P2 bases are shown in Figs. 14 and 15, respectively. The improvements of the CBC treatment are significant and the third order accurate solution of Fig. 15 agrees with the analytic result to plotting accuracy.

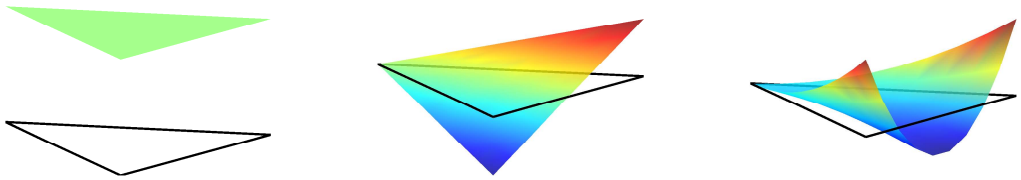


(a) (b) (c)

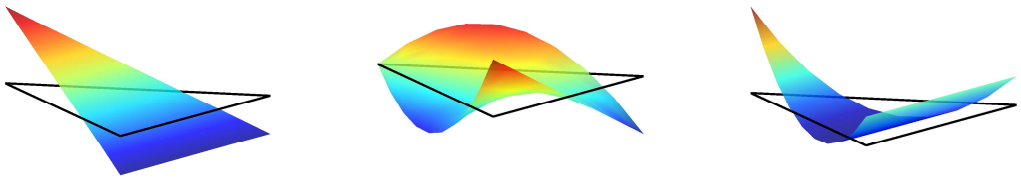
(d) (e) (f)

(g) (h) (i)

Figure 4: Basis functions over the standard quadrilateral region: (a) $(p, q) = (0, 0)$, (b) $(p, q) = (1, 0)$, (c) $(p, q) = (2, 0)$, (d) $(p, q) = (0, 1)$, (e) $(p, q) = (1, 1)$, (f) $(p, q) = (2, 1)$, (g) $(p, q) = (0, 2)$, (h) $(p, q) = (1, 2)$, (i) $(p, q) = (2, 2)$.



(a) (b) (c)



(d) (e) (f)

Figure 5: Basis functions over the standard triangular region: (a) $(p, q) = (0, 0)$, (b) $(p, q) = (1, 0)$, (c) $(p, q) = (2, 0)$, (d) $(p, q) = (0, 1)$, (e) $(p, q) = (1, 1)$, (f) $(p, q) = (0, 2)$.

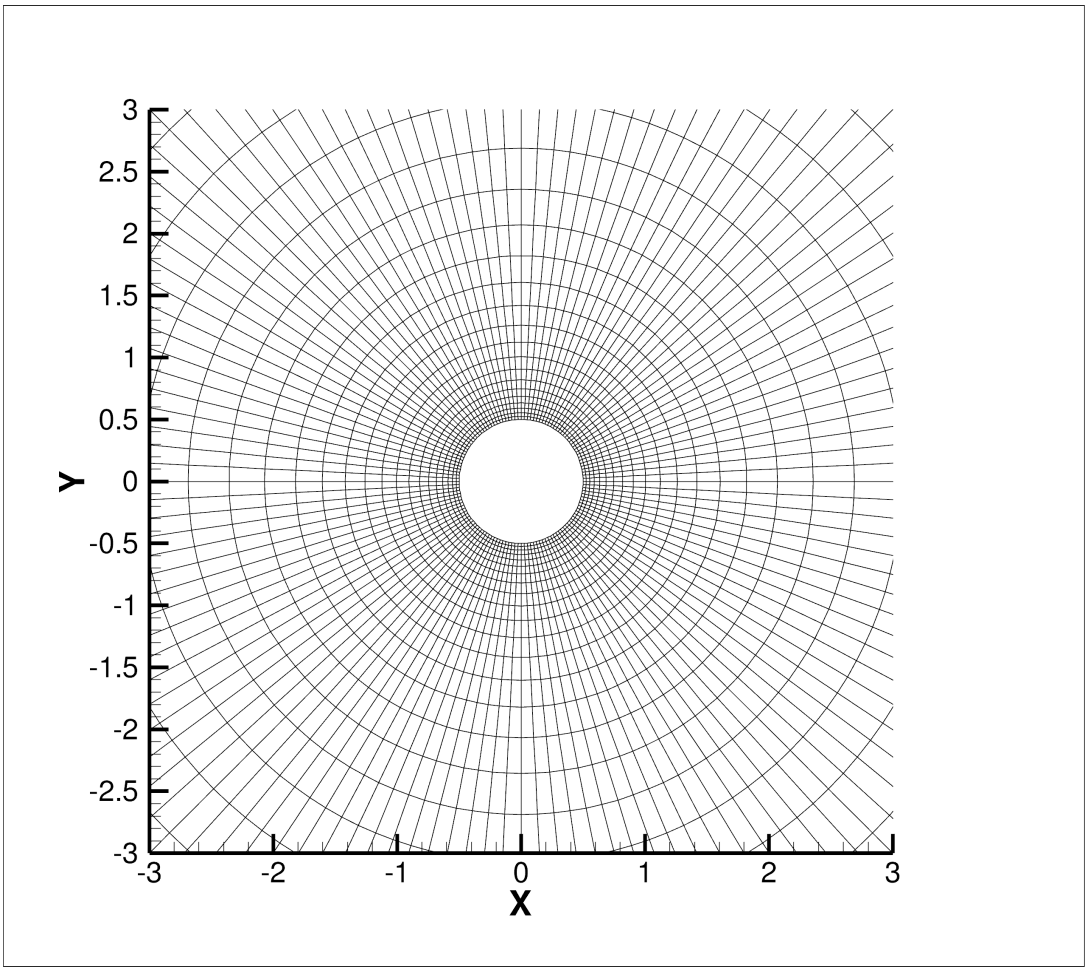


Figure 6: Structured mesh around a cylinder.

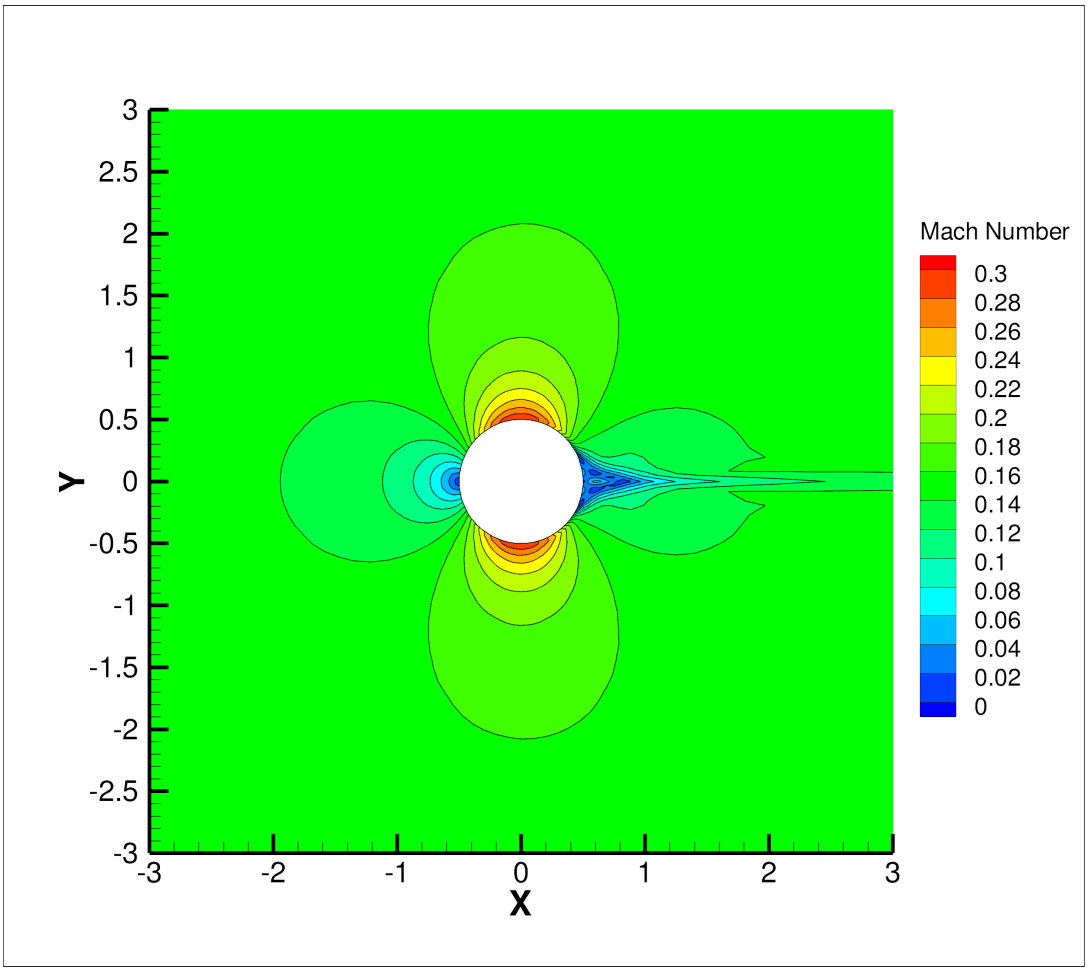


Figure 7: Mach contours with no CBC (P1) for the structured mesh.

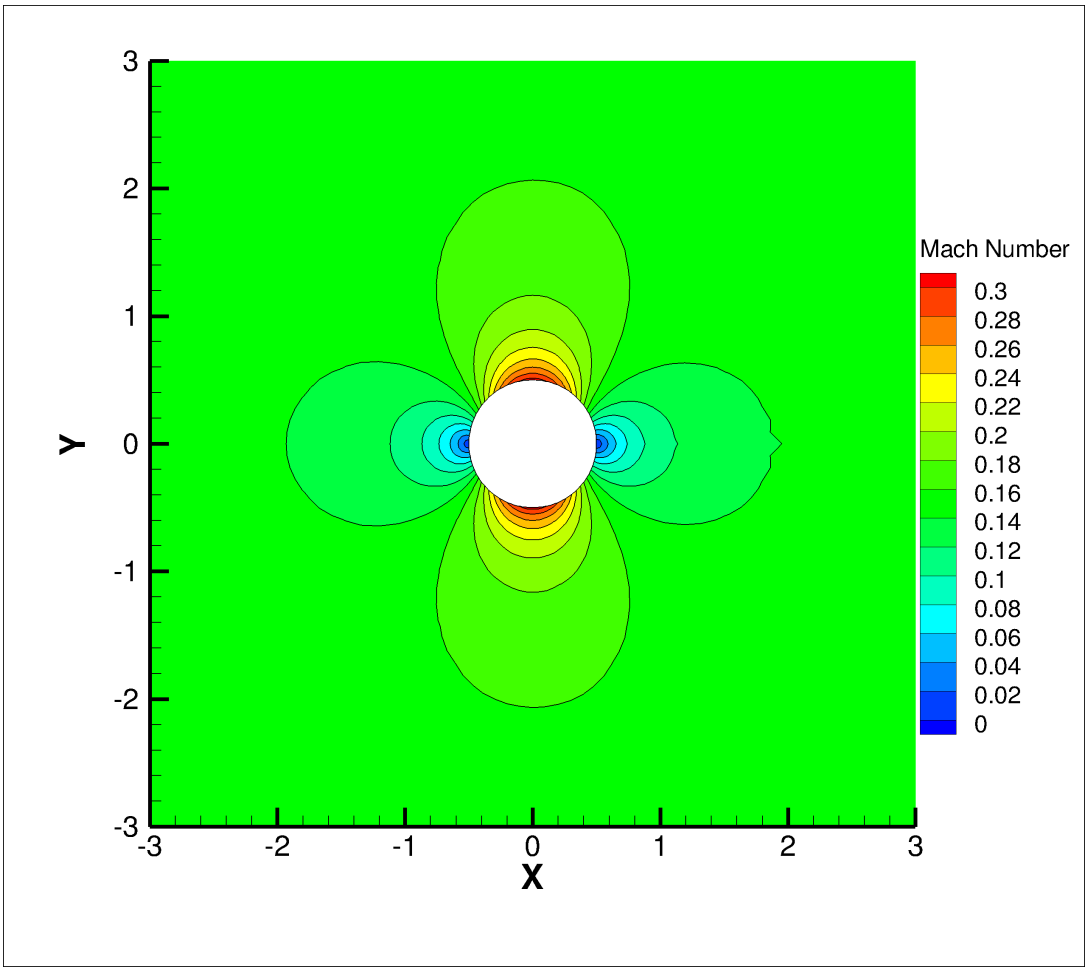


Figure 8: Mach contours with CBC (P1) for the structured mesh.

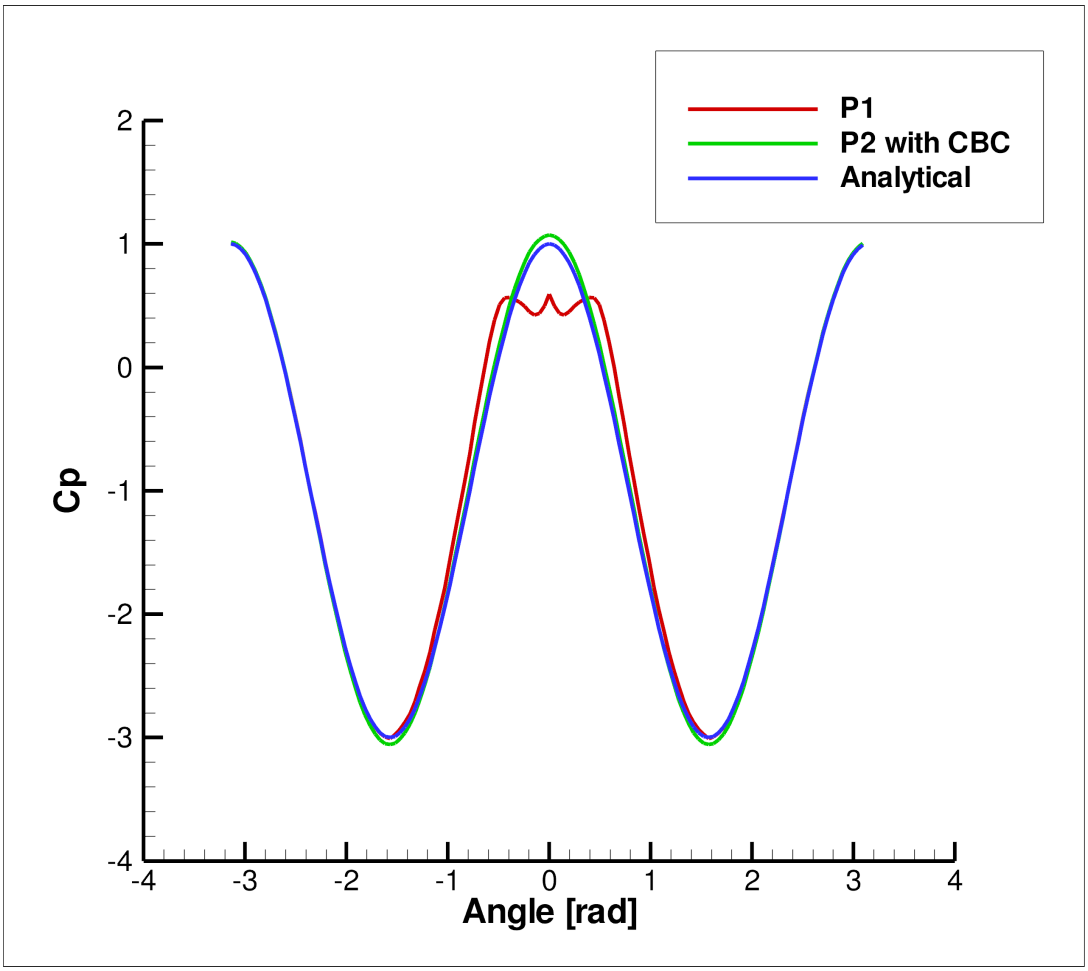


Figure 9: Line plots of computed and analytical pressure coefficient for the structured mesh.

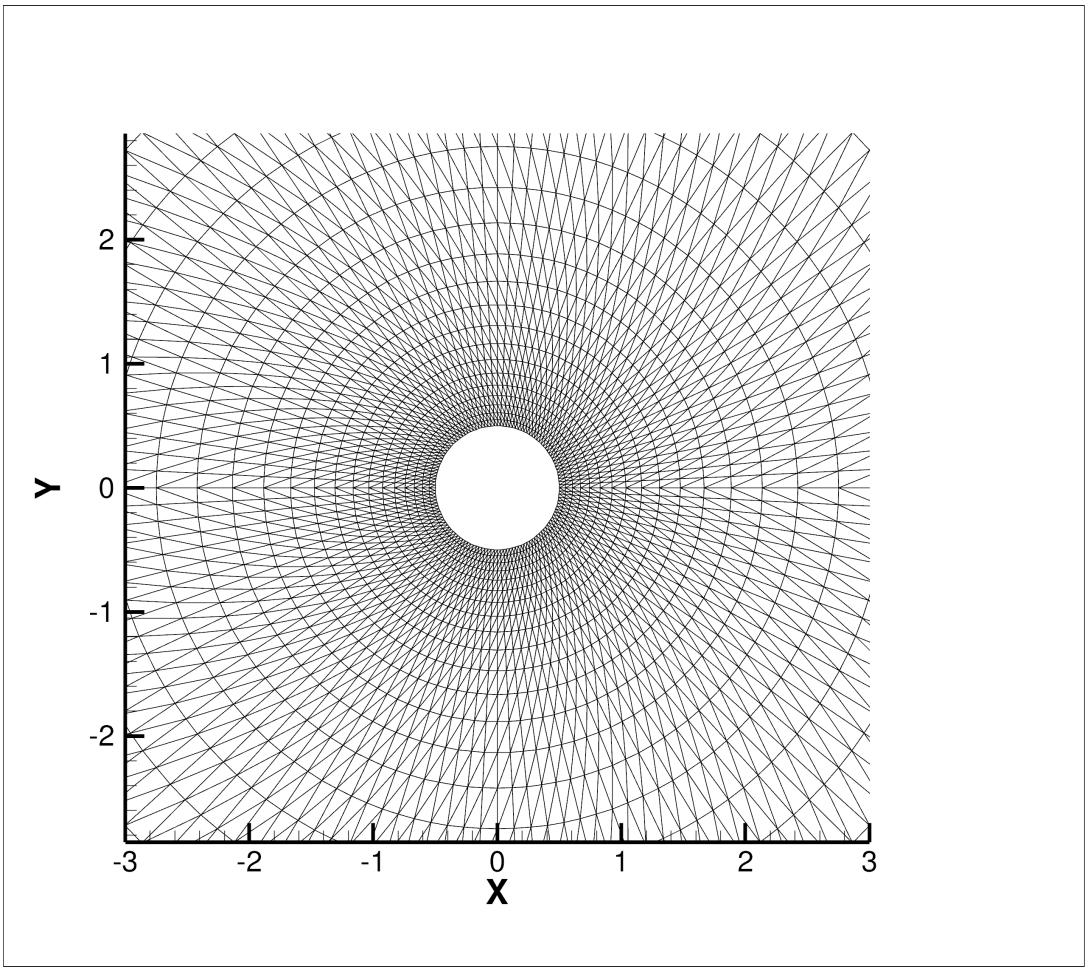


Figure 10: Unstructured mesh around a cylinder for the unstructured mesh.

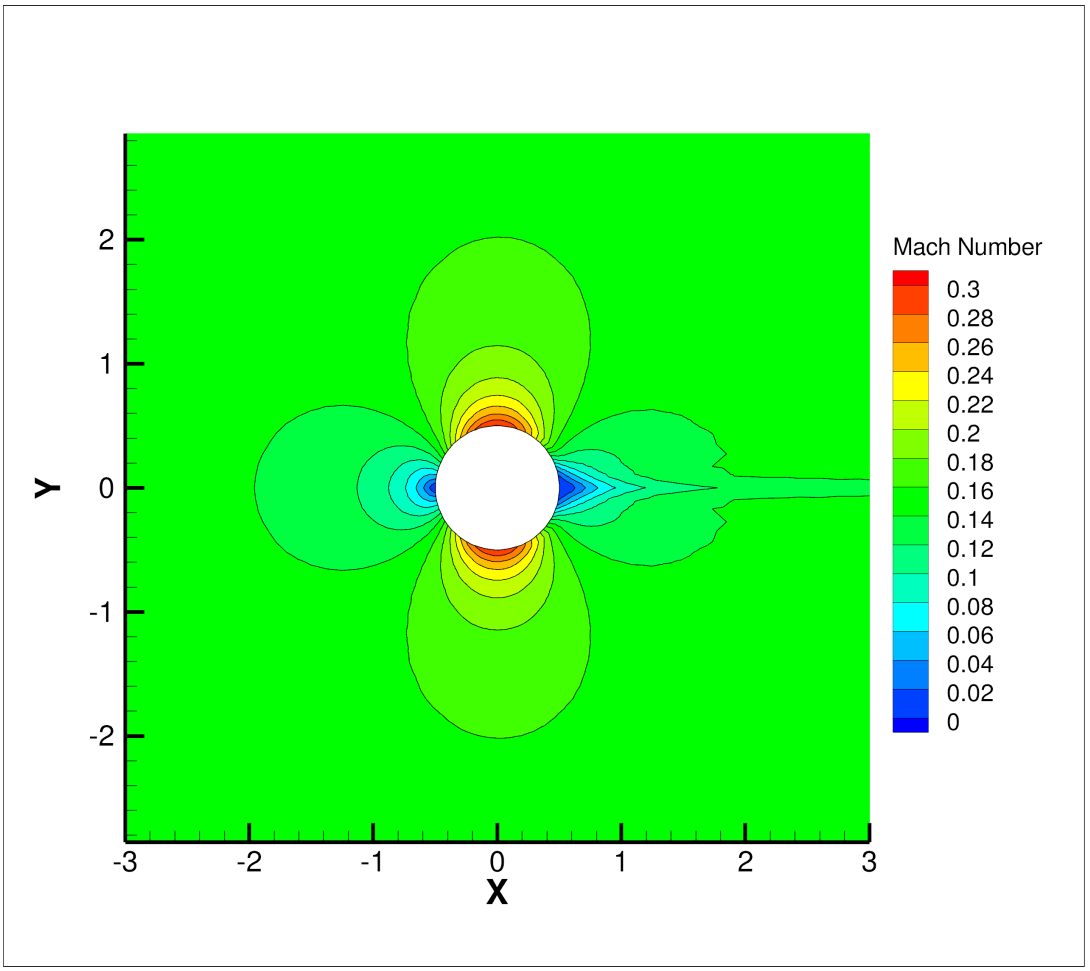


Figure 11: Mach contours with no CBC (P1) for the unstructured mesh.

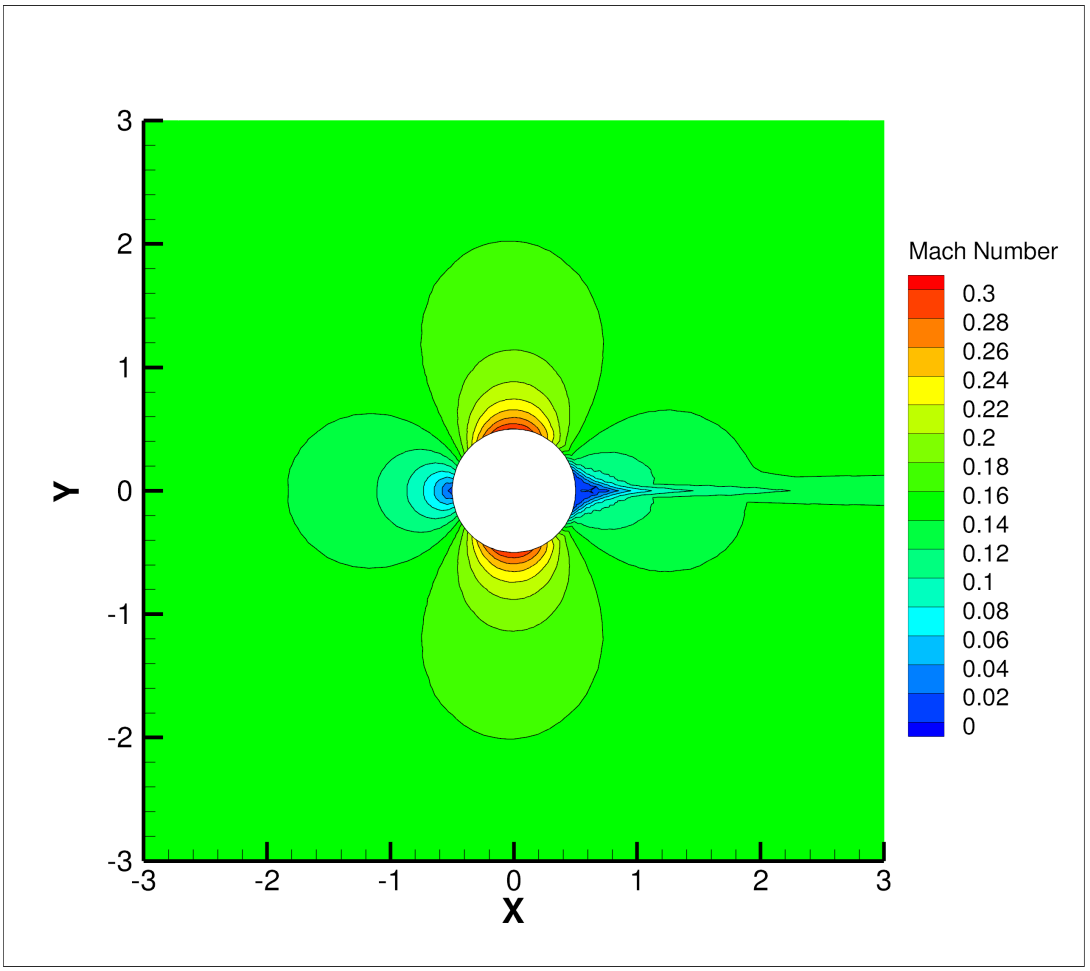


Figure 12: Mach contours with no CBC (P2) for the unstructured mesh.

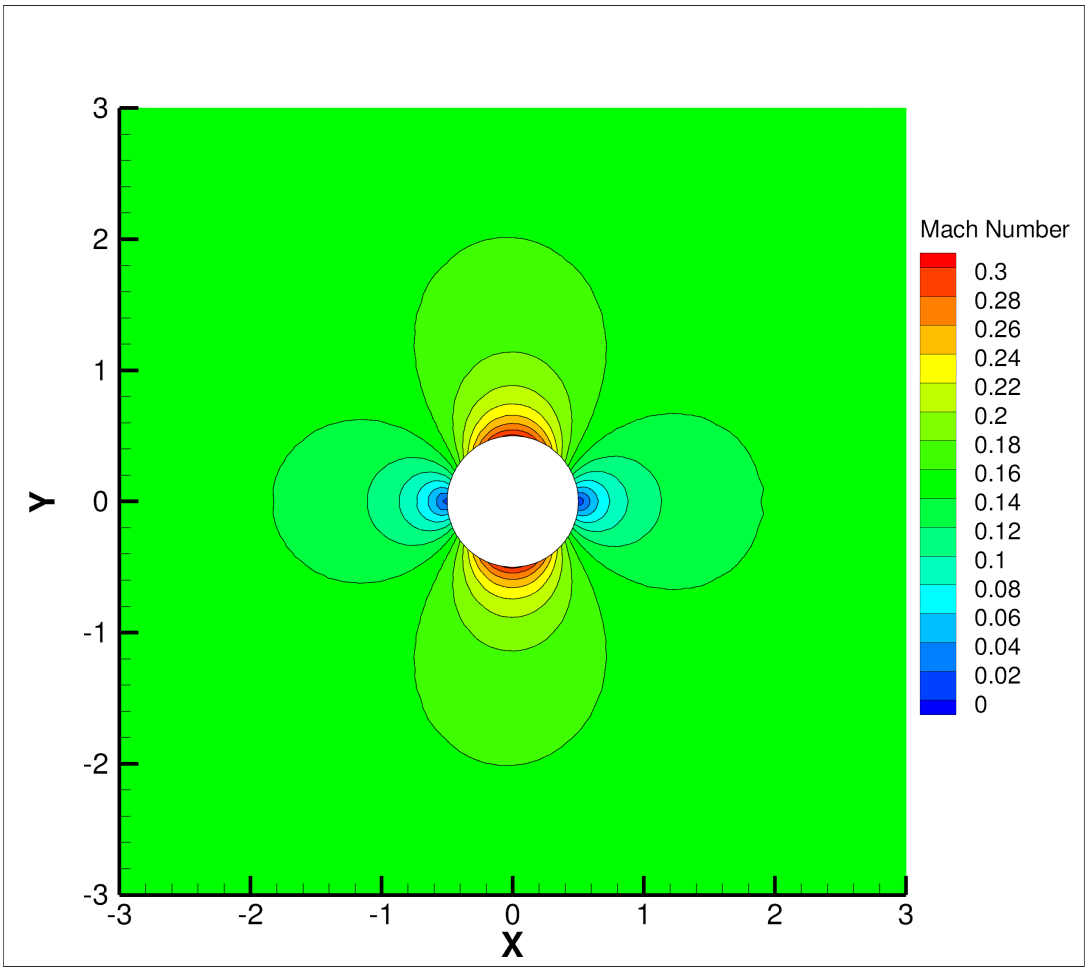


Figure 13: Mach contours with CBC (P2) for the unstructured mesh.

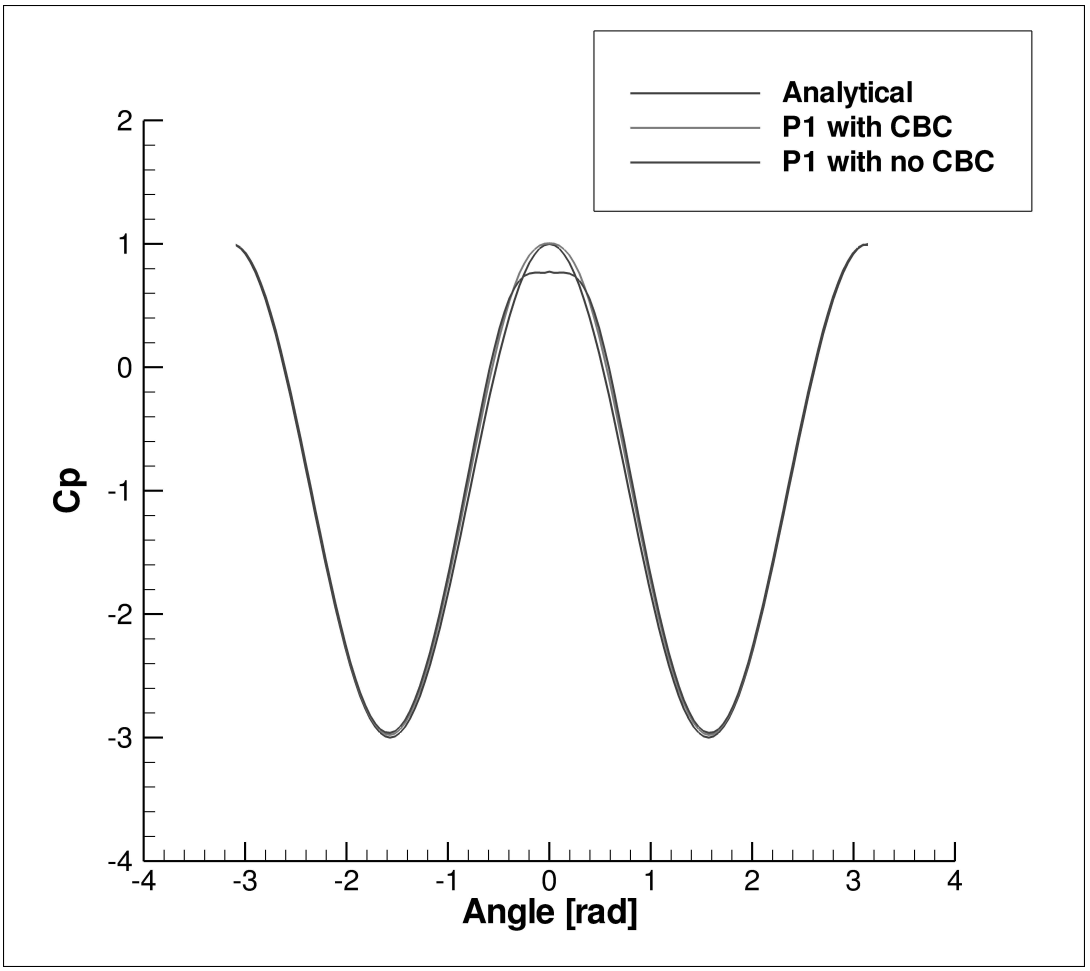


Figure 14: Line plots of computed and analytical pressure coefficient for the unstructured mesh (P1).

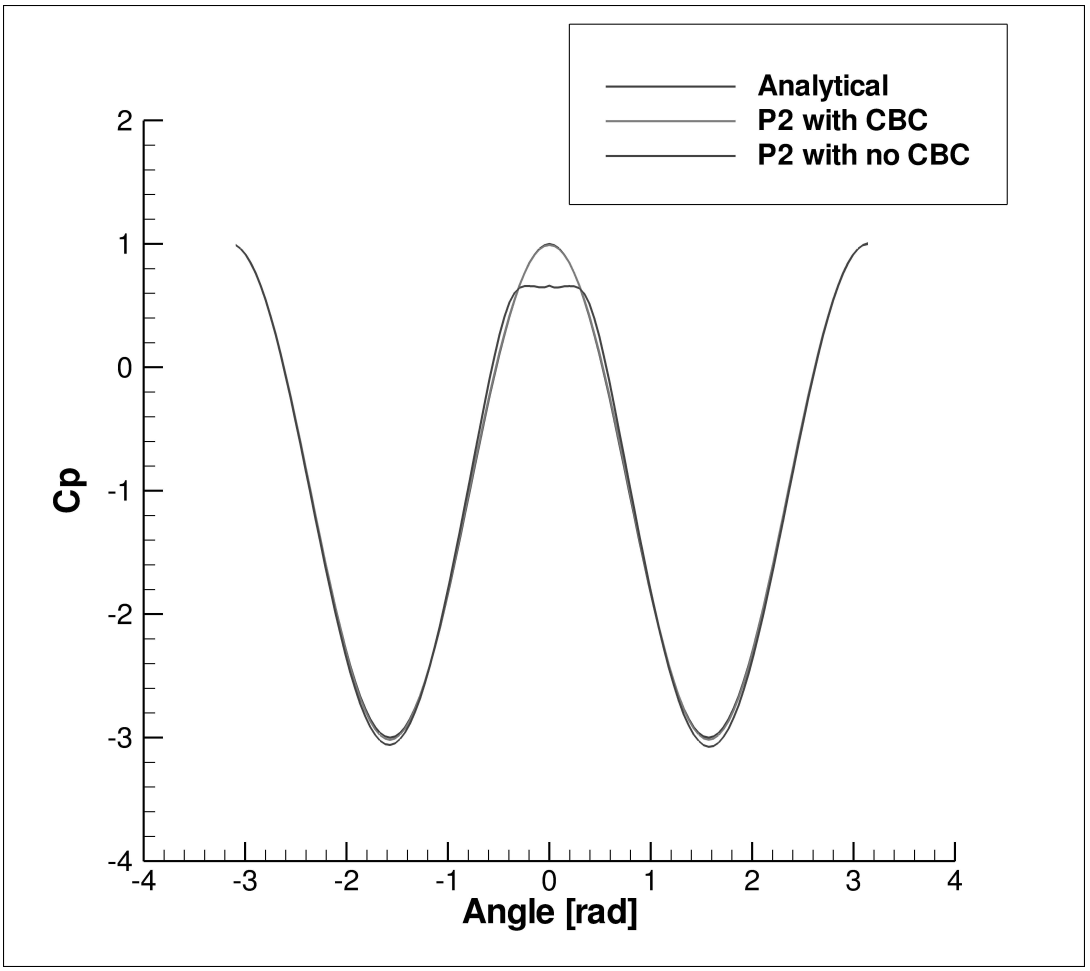


Figure 15: Line plots of computed and analytical pressure coefficient for the unstructured mesh (P2).

References

[1] C. Hirsch. *Numerical Computation of Internal and External Flows*, volume 2. John Wiley & Sons, 1988.

[2] G.E. Karniadakis and S.J. Sherwin. *Spectral/hp Element Methods for CFD*. Oxford Univeristy Press, 2nd edition, 2003.

[3] T.C. Warburton, S.J. Sherwin, and G.E. Karniadakis. Basis Functions for Triangular and Quadrilateral High-Order Elements. *SIAM Journal of Scientific Computing*, 20(5):1671–1695, 1991.

[4] T.C. Warburton. *Spectral/hp methods on polymorphic milti-domains: Algorithms and Applications*. PhD thesis, Brown University, 1998.

[5] B. Landmann. *A parallel discontinuous Galerkin code for the Navier-Stokes and Reynolds-averaged Navier-Stokes equations*. PhD thesis, University of Stuttgard, 2008.

[6] Lilia Krivodonova and Marsha Berger. High-order accurate implementation of solid wall boundary conditions in curved geometris. *Journal of Computational Physics*, Vol. 211:pp. 492–512, 2006.

Quarterly Progress Report for the
EOARD Project

Development of High–Order Methods for Multi–Physics Problems Governed by Hyperbolic Equations

by

John A. Ekaterinaris

FORTH / IACM

Award No. FA 8655–03–1–3085

Reporting period June 1 – August 31, 2009

Summary

The discontinuous Galerkin method for the two-dimensional Euler equations was extended to mixed-type (triangular-quadrilateral) meshes. Numerical experiments demonstrated that the design accuracy level is retained for both type of meshes separately and for the global solution with mixed-type meshes. Furthermore, the capability of numerical solutions with p-type adaptivity on fixed meshes was implemented. Numerical results demonstrate that significant savings in computing time can be obtained by using mixed-type meshes and p-adapted numerical solutions. These savings are expected to be even larger for viscous flows.

Mixed-type mesh and p-adaptivity

The numerical method, which was described in detail in the previous quarterly report, it has the following features:

1. It is implemented in the computational space.
2. Uses hierarchical tensor product Jacobi polynomial basis functions.
3. Performs numerical integration of the volume and line integrals that appear in the DG methods on the unit square.
4. The basic element in the transformed space is the unit square and triangular elements are represented via the collapsed coordinate system.

These features make the method general, easy to extend to three dimensions, enable discretization on mixed-type meshes (because all operations for both triangles and quadrilaterals of the physical space are performed for the same unit square), and use of arbitrary p-adapted numerical solutions because the tensor product basis functions on the unit square (for both triangles and quadrilaterals of the physical space) are hierarchical. All these features were exploited to solve test problems on mixed-type meshes and demonstrate the computational saving of p-adapted solutions. In this report, p-adapted solutions are demonstrated only for artificially segregated meshes. For example near to the cylinder region of mesh over a cylinder is flagged as a region where the solution is performed using P2 polynomial bases (third-order accurate solution) while in the rest of the domain the numerical solution is obtained using second order accuracy.

For the general case it is desirable to be able to increase the polynomial approximation locally bases on criteria resulting from the physics of the computed solution. In order to establish procedures for p-adaptive numerical solutions on fixed meshes based on the computed flow features we the following example is considered. The following form of the Euler equations is solved on a triangular mesh

$$\frac{\partial}{\partial t}\begin{Bmatrix}\rho\\ \rho u\\ \rho \upsilon\\ \mathrm{E}\end{Bmatrix}+\frac{\partial \mathbf{f}}{\partial x}+\frac{\partial \mathbf{g}}{\partial y}=\begin{Bmatrix}0\\ 0\\ \rho\\ \rho \upsilon\end{Bmatrix}$$

The source terms generate the Taylor-Rayleigh instability[1, 2] with the initial and boundary conditions shown in Fig. 1

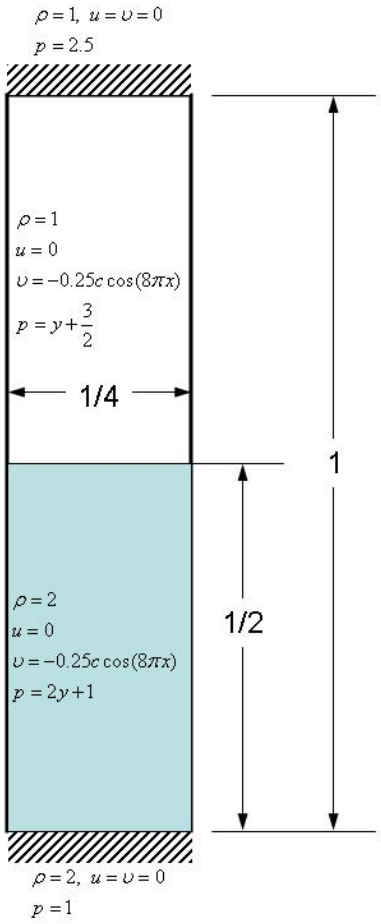


Figure 1. Initial and boundary conditions for the Taylor-Rayleigh instability

The heavy gas is on the bottom, the normal velocity component is initially perturbed, and the linearly varying pressure is continuous at the interface. Numerical solutions for this problem performed with WENO schemes[3] or the DG method for increasing order of accuracy on fixed meshes with size $h = 1/240, 1/480, 1/960, 1/1920$ or $h = h_o,\ h_o/2,\ h_o/4,\ h_o/8$ have demonstrated that with the increase of resolution (h refinement or p increase) results in capturing of finer and finer flow features. We attempt to resolve fine feature with local p-type refinement only at the regions of steep flow gradients on fixed h_o size meshes. Currently steep density gradients are detected and the polynomial approximation in these regions is increased, for example P8 approximations of the approximate solution are used for the maximum of the density gradient. At neighbouring to P8 cells P7 approximation is used and the order of approximation progressively diminishes while we move away from regions with large density gradients. The order of approximation increases or drops according to the magnitude of density gradient as the flow develops. The full locally p-adaptive scheme and the computed solutions will be shown in the next report. Similar approach will be used for the accurate and efficient computation of viscous flow features and flows with shocks,

Results

The p-adaptive capabilities on mixed-type meshes are demonstrated for the computation of inviscid flows over a cylinder and a NACA-0015 airfoil. The mixed-type mesh used for the computation of the flow over the cylinder is shown in Fig. 2. The near field is computed on a quadrilateral mesh and the far field is computed on a triangular mesh. On the quadrilateral mesh any type of approximation can be used. In this example, the order of accuracy in both the quadrilateral and triangular meshes was fixed. An example where the inner solution was computed with P2 basis functions (third order accurate) and the outer solution on the triangular mesh is second-order accurate (P1 polynomials) is shown in Fig. 3. The computed pressure coefficient at $M_\infty = 0.15$ is compared with the exact value for incompressible inviscid flow over a cylinder in Fig. 4. For this computation the boundary of the cylinder was approximated with linear elements and the boundary treatment proposed by Krivodonova and Berger[4] was used, which yield effectively quadratic approximation of the boundary. Incorporation of solid the boundary representation of Ref. 4 significantly increases the accuracy of the numerical solution and almost eliminates artificial numerical entropy layers that are generated from low order representation of the curved boundary.

Next, the inviscid flow over a NACA-0012 at low incidence $\alpha = 2$ deg. and $M_\infty = 0.4$ was computed on a mixed type-mesh. The numerical mesh is shown in Fig. 5. The computed pressure and Mach number contours are shown in Fig. 6 and 7, respectively. The gain of using mixed-type meshes and p-adaptivity is more pronounced for computation of viscous flows. Currently, implementation of viscous terms for the DG method is underway. The narrow stencil implementation of viscous terms for the DG method and results from computations of viscous laminar flows over the cylinder and the airfoil will be presented in the next quarterly report.

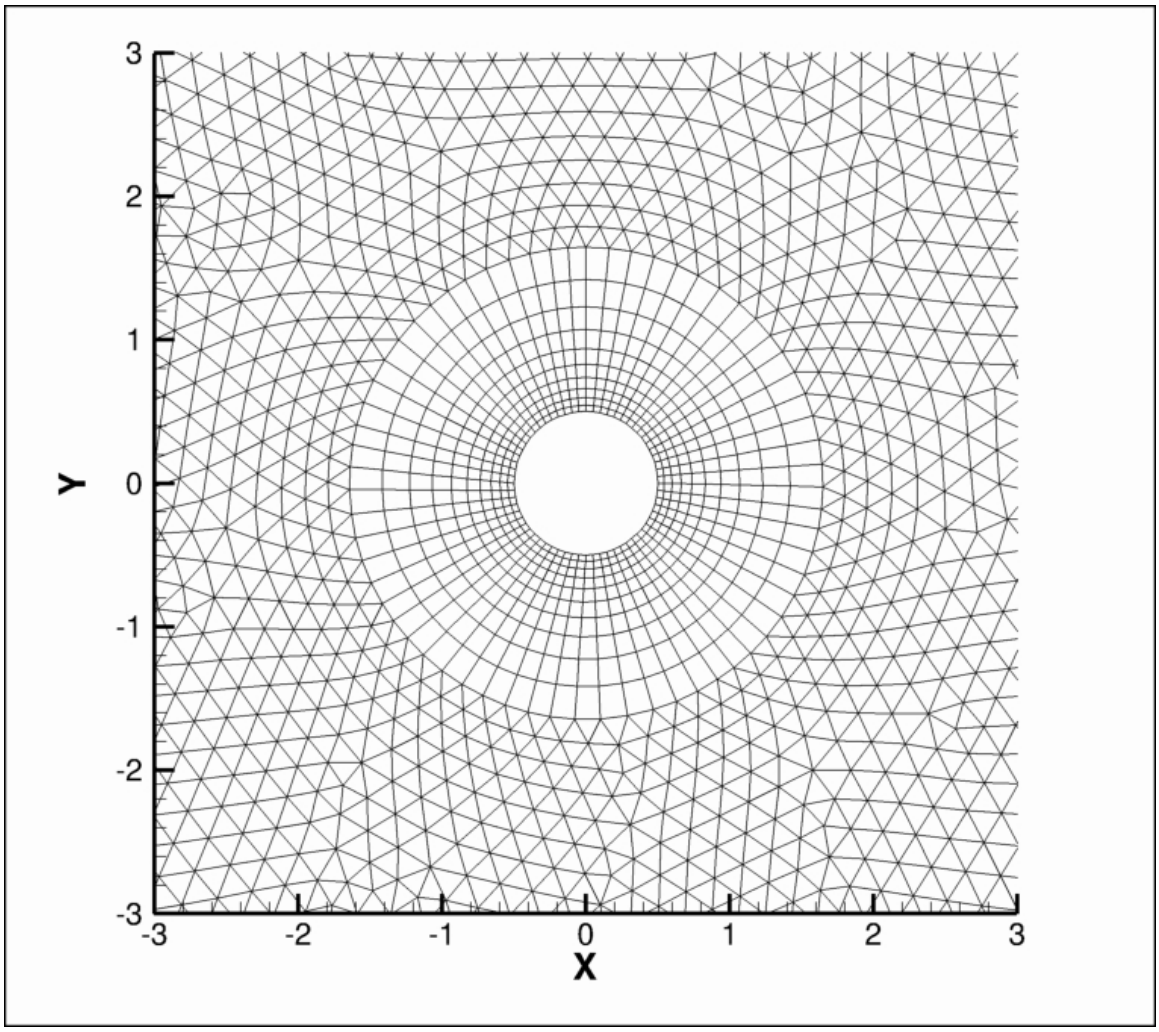


Figure 2. Mixed-type mesh for the computation of inviscid flow over the cylinder.

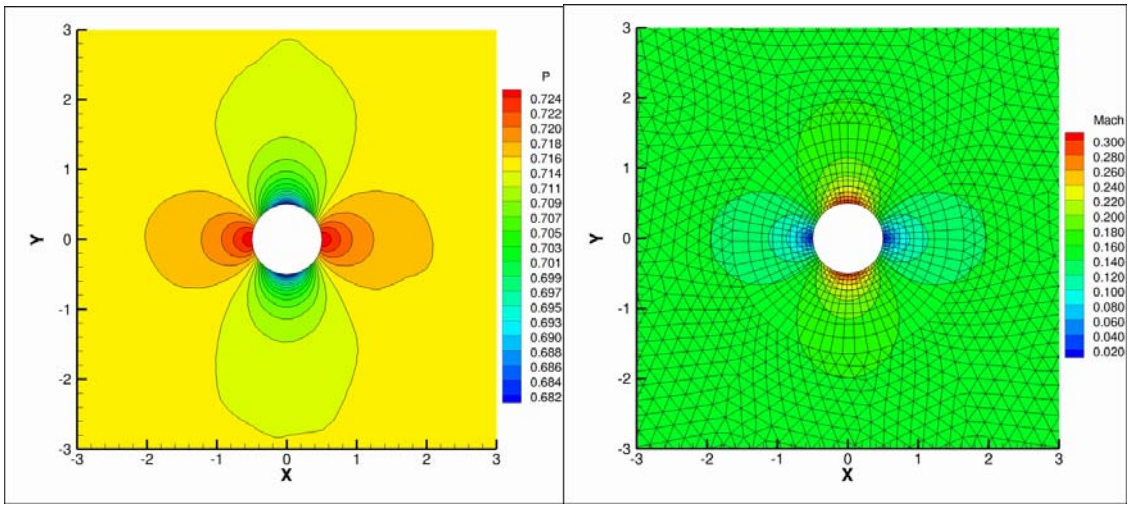


Figure 3. Computed pressure and Mach distribution at $M_\infty = 0.15$.

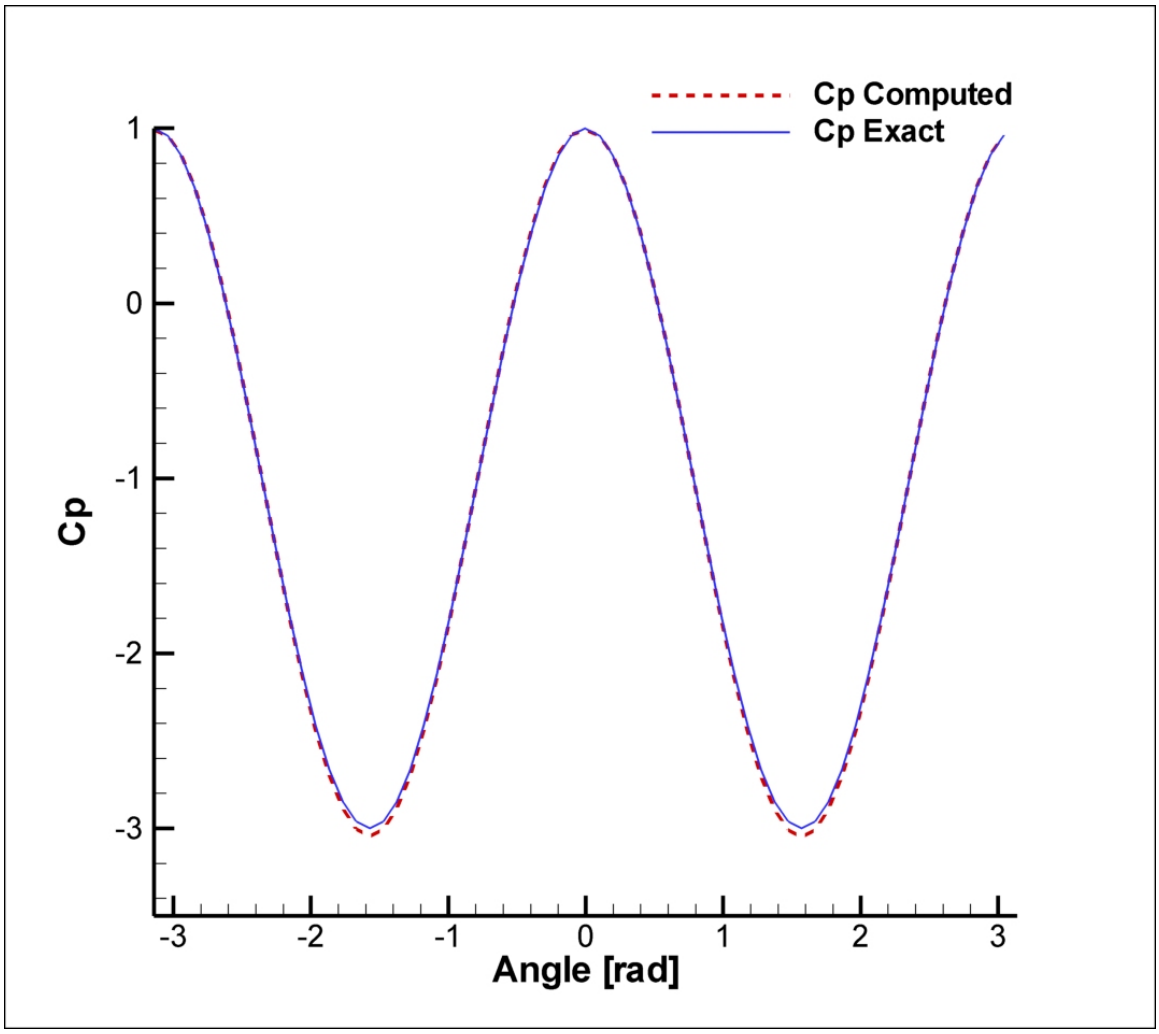


Figure 4. Comparison with the exact solution.

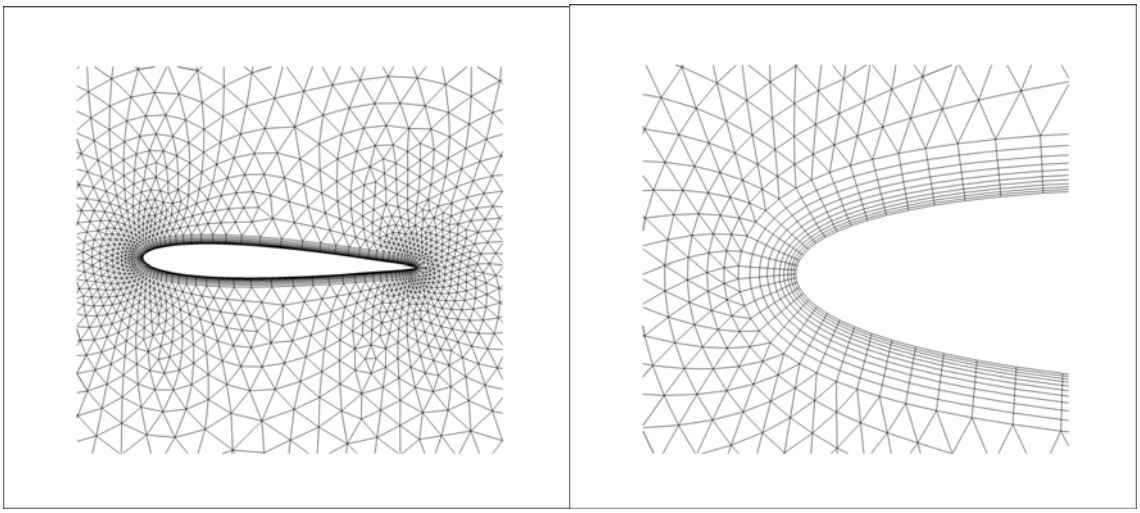


Figure 5. Mixed-type mesh for the computation of inviscid flow over the NACA-0012 section.

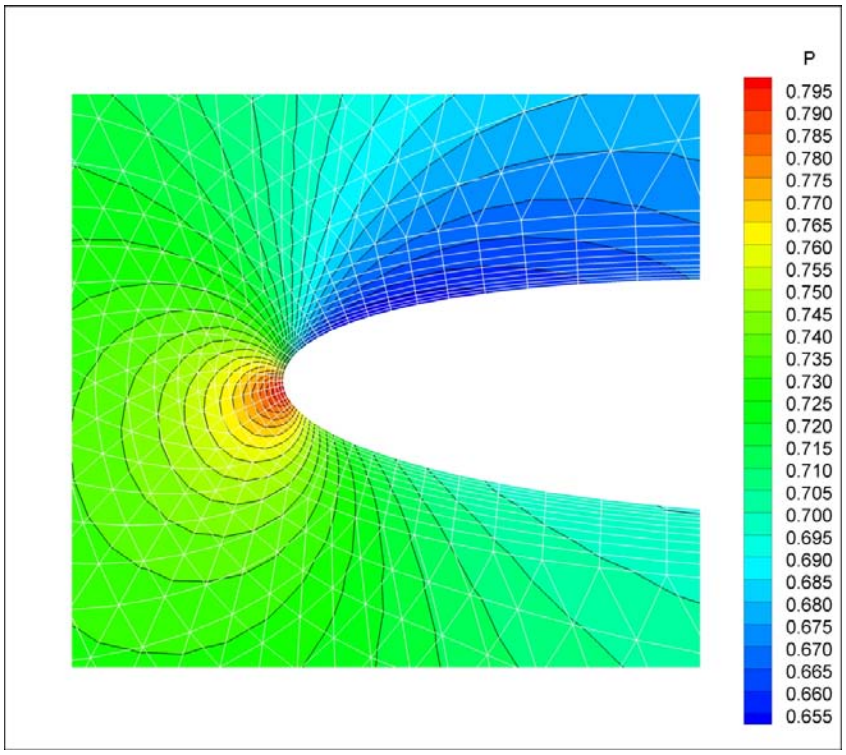


Figure 6. Computed pressure distribution at $M_\infty = 0.4$.

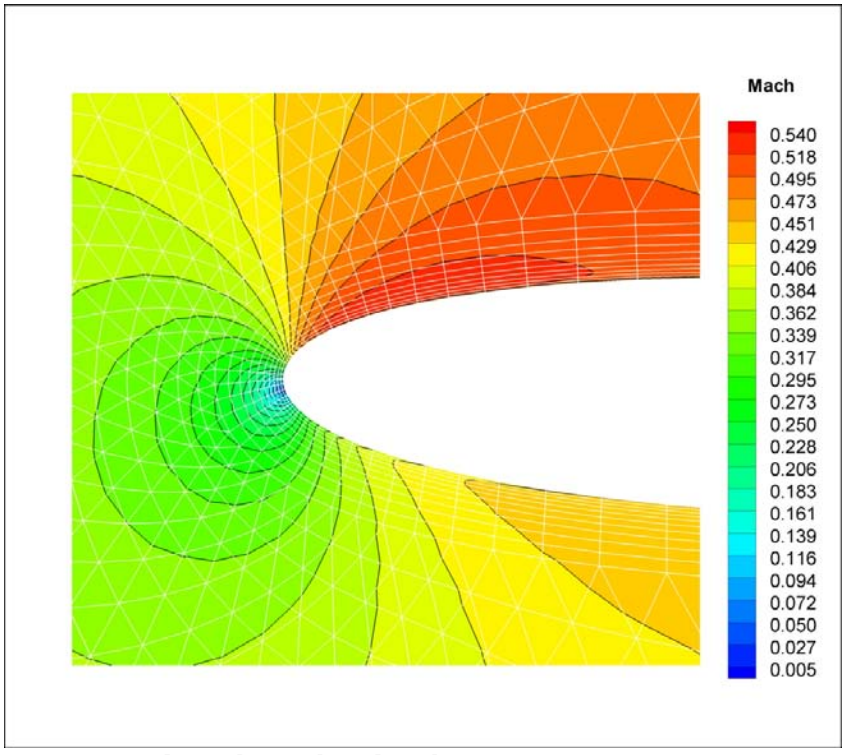


Figure 7. Computed Mach number distribution.

References

1. J. Glimm, J. Grove, X. Li, W. Oh, and D.C. Tan, “The dynamics of bubble growth for Rayleigh–Taylor unstable interfaces,” Physics of Fluids, Vol. 31, 1988, pp. 447–465.

2. Y.-N. Young, H. Tufo, A. Dubey, and R. Rosner, On the miscible Rayleigh–Taylor instability: two and three dimensions, *Journal of Fluid Mechanics*, Vol. 447, 2001, pp. 377–408.

3. J. Shi, Y.-T. Zhang, and C.-W. Shu, “Resolution of high order WENO schemes for complicated flow structures,” *Journal of Computational Physics*, Vol. 186, No. 2, 2003, pp. 690-696.

4. L. Krivodonova and M. Berger, “High-order accurate implementation of solid wall boundary conditions in curved geometries,” *Journal of Computational Physics*, Vol. 211, No. 2, 2006, pp. 492-512.

Quarterly Progress Report for the
EOARD Project

Development of High–Order Methods for Multi–Physics Problems Governed by Hyperbolic Equations

by

John A. Ekaterinaris

FORTH / IACM

Award No. FA 8655–03–1–3085

Reporting period January 1 – March 31, 2010

Acceleration of Large-Scale Simulation Using Many-Core Architectures

John A. Ekaterinaris
and
Konstantinos I. Karantasis

Summary

Current trends on high performance computing are moving towards the deployment of several cores on the same chip of modern processors in order to achieve substantial execution speedup through the extraction of the potential fine-grain parallelism of applications. At the forefront of this trend we find nowadays the modern Graphics Processors Units (GPUs), which due to their simplistic design are able to encompass hundreds of independent processing units on a single chip in contrast to their respective CPUs, which at the moment include only a few cores on the same chip. In order to study the potential speedup of computationally intensive applications that utilize the many-core architecture of GPUs, this report presents a highly accelerated implementation of the finite-difference weighted essentially non-oscillatory (WENO) scheme and the discontinuous Galerkin finite element method. These discretization techniques are suitable for direct numerical simulations (DNS) large eddy simulations (LES) of compressible turbulence, however, they require large computing resources in order to achieve high Reynolds numbers. Our implementation targets on large-scale simulations using the CUDA parallel programming and constitues a paradigm of GPU's applications in CFD. The results of the current implementation demonstrate that such a computationally intensive application could be highly accelerated running on the NVIDIA Tesla C1070 many-core GPU.

1 Background

Numerical simulation of transition and turbulence in high-speed flows is daunting because of the difficulty in ensuring high-resolution and fidelity in capturing small disturbances in an environment containing sharp gradients associated with shocks and relatively thin boundary layers. Even with use of higher-order approaches many shock capturing methods introduce spurious (numerical) noise which contaminates the solution beyond acceptable limits, which can lead to significant damping of turbulence fluctuations, and masks the effects of the subgrid-scale (SGS) models. Specification of boundary conditions ensuring that the numerical discretizations remain stable is also a critical issue. Robust, high-fidelity and accuracy methodologies that are capable of treating complex flows and are applicable for high-resolution numerical solutions in complex domains are therefore solemnly required.

The objective of the present work is to apply conservative, high-order accurate, shock-capturing methods that are suitable for the simulation of supersonic flows in domains with moderate complexity for massively parallel architectures. A high order finite difference weighted essentially nonoscillatory (WENO) scheme and the DG method are used for the numerical solution of the governing equations. The main disadvange of WENO discretization is that with the increase of the order of accuracy increases the width of the computational stencil. As a result, parallelization with traditional domain decomposition methods could become inefficient. On the other hand the in general inherently unstructured data access of the DG method requires use of complex domain decomposition procedures. In this work we only use structured data access and demonstrate acceleration with the use of graphics processors units (GPUs).

The implementation of the accelerated finite difference WENO scheme that we present here is based on the CUDA parallel programming model [1]. CUDA comprises a programming environment that provides an extention to the C programming language accompanied by the necessary libraries to support the execution of code on top of the NVIDIA GPUs. Using CUDA the program must be structured on distinct portions which are called kernels and are destined to be executed on the side of the GPU. The execution of kernels follows the programming paradigm of SIMT (Single Instruction Multiple Threads) which resembles the traditional programming model of vector processors, the SIMD (Single Instruction Multiple Data) paradigm. Nevertheless SIMT is less restrictive than SIMD and allows programmers to write either data-parallel code as in the case of SIMD or arbitrary thread-based parallel code. In order to initiate a kernel execution using CUDA a mapping of application threads into blocks and accordingly a mapping of these blocks on a grid must be provided by the programmer.

Data parallel applications are serious candidates among the applications that have the potential to achieve a significant acceleration through the utilization of many-core GPUs. Similar efforts have recently taken place in the area of scientific and recent demonstrations in CFD concern simulations on both structured [2] and unstructured [3], [4] meshes. The work that we present here proves that CFD applications with similar characteristics can benefit from heterogeneous computing systems that involve CPU and GPU co-processing.

2 Acceleration on Graphics Processors

In thus section the procedure followed to reach the code that was able to parallelize on GPUs is described. The steps comprising this procedure starting from the FORTRAN version to the final CUDA version are given in detail. Our approach for GPUs parallelization is still based on traditional domain decomposition techniques therefore domain decomposition is described first.

2.1 Domain decomposition

Similar to other computational techniques used in the CFD community, the high order WENO scheme for the numerical solution of the Euler and Navier-Stokes equations, has been initially implemented in FORTRAN-90 for single block structured meshes.14 Parallelization of the code for clusters was subsequently achieved through the use of the message passing protocol MPI and domain decomposition. On the MPI parallel version, the domain decomposition was performed by subdividing the domain only along the axial direction. In order to study the potential acceleration of the code on heterogeneous schemes that involve CPU and GPU co-processing we decided to port our application on the C language although the first parallelizing compiler for CUDA and Fortran have been released by the time that our research was taking place [5]. According to that decision, the FORTRAN-90 parallelized version of the code formed the basis for the shared memory parallel implementation on C/OpenMP that we evaluate in this paper. The decomposition of the computational domain for OpenMP multi-threaded execution is depicted in the schematic of Fig. 1. The generalized coordinates form of the governing equations is solved and the global computation of dimension *Imax* x *Jmax* x *Kmax* is subdivided along the *i ,* or direction, which is often the streamwise direction, in N subdomains of dimension $\{Imax/N\}+2m$ x *Jmax* x *Kmax,* where m is the number of the ghost points required for data transfer. The number of ghost points varies with the order of the base scheme and for the 5th order WENO is $m=3$, while for the 9th order WENO is $m=5$.

The C/OpenMP version of the WENO solver was validated by comparing results of the baseline FORTRAN-90 code for several cases for both single processor solutions and solutions with domain decomposition. Next parallel implementation on GPUs was implemented and the C language version was ported into CUDA.

2.2 GPU processing

In the presence of considerably larger number of cores on the GPU as opposed to the CPU, in order to fully utilize the afforded computational resources, we extended the domain decomposition along a second direction. In that sense every element on the 2D plane that is mapped on the GPU is assigned on a different thread and corresponds to the processing of a group of points along the third axis. This strategy of a 2D domain decomposition that is depicted in Fig. 2 and is exercised both on the single GPU mode and the multi-GPU mode adds additional capabilities for large scale DNS or LES simulations where large number of points is used along the streamwise and normal to the wall directions while the number of points in the spanwise direction, which is often considered periodic, is smaller. Moreover, to maximize the throughput with appropriate expression of parallelism, the number of threads on each block that handles the respective points has been set as a multiple of 32, which corresponds to the warp size of the GPU. This was decided because the warp is the minimum unit

that can be scheduled at once on a single stream multiprocessor of the GPU.

Along with the need to express a high degree of parallelism on our scheme, the parameter that we considered and has a great impact on performance is the memory utilization, especially in the case of multi-GPU execution. For that reason, firstly, we had to minimize data transfers between the host memory and the device memory. Therefore, in the presented implementation all the computations that are required by the Runge Kutta time stepping for updating the numerical solution have been assigned to the GPU, even though some of the computations do not exhibit a high degree of parallelism.

Subsequently on the side of each GPU, in order to utilize appropriatelly the memory hierarchy of the device, we use the texture memory space for the placement of read-only data that are computed before the time evolution of the simulation begins. This happens in order to benefit from the caching mechanism of the texture memory, which is by design optimized for spatial locality. The results that are computed on each time step are stored in global – read/write – memory. In addition, due to the specification of CUDA architecture that forbids communication between threads that reside on distinct blocks, each block computes it's ghost points instead of exchanging them with neighboring blocks.

Concerning the necessary synchronization that is imposed by the computation scheme, it is restricted on barrier synchronization between the threads of the same common block inside each CUDA kernel. In that way we signify the update of auxiliary local variables required for Roe's averaging and other local arrays for the right and the left eigenvectors that are evaluated at the average state. The need for mutual exclusion is minimized on the atomic max operation that is used in order to compute the maximum eigenvalue required for the construction of the Lax-Friedrisch numerical flux.

3 Results

In this section the cases chosen for the validation of the parallelization procedure and evaluation of performance are presented. Next the performance is evaluated for both inviscid and viscous flow computations. Two examples were used to evaluate the performance of the current GPU implementation of the high order numerical method. Oblique inviscid shock reflection, and simulation of Rayleigh–Taylor instability. [6], [7] The code is three dimensional and numerical simulations for two dimensional cases were performed by using appropriate to the scheme order number of planes in third dimension. Oblique shock reflection from a solid surface is a classical compressible code validation case while the interaction of an oblique shock with a boundary layer is a problem of current interest where LES simulations could shed light to the dynamics of the interaction. The GPUs parallelized high order method was tested for this problem first. Numerical solutions for a sequence of meshes of different sizes were obtained in order to evaluate the performance on GPUs.

The computational domain for the simulations of the Rayleigh–Taylor instability is the box (1×0.25) in two dimensions and $(1 \times 0.25 \times 0.25)$ in

three dimensions. The "heavy" fluid is on the left site and has density $\rho_L = 2$ while the "light" fluid on the right has density $\rho_R = 1$. The interface is between the two fluids is at $x = 1/2$ and the variation of the initial pressure is linear throughout the domain. The initial pressure in the domain of the heavy fluid on the left is $p_L(x) = 1 + 2x$, while the variation of the pressure on the right is $p_R(x) = 1.5 + x$. The initial perturbation of axial velocity $u(y) = -0.025\cos(8\pi y)$ is specified throughout the computational domain and the source term,16, 17 $S = (0,\ \rho,\ 0,\ 0,\ \rho u)^T$, is added for both viscous and inviscid simulations.

The significant reductions of the computational time achieved through the use of GPUs, which are discussed in detail in the next section, made possible simulations of the Rayleigh–Taylor instability with different mesh densities and schemes of different order of accuracy. A comparisons of two dimensional simulations obtained on a series of meshes is shown in Fig. 3. All simulations of Fig. 3 are for the same final time. Clearly increase of the order of accuracy yields the same effect as doubling of the mesh density in both directions. Furthermore, use of GPUs made possible three dimensional simulations. An example of a three dimensional simulation obtained on a relatively coarse 240 x 60 x 60 point mesh is shown in Figs. 4 and 5.

3.1 Performance evaluation on Graphics Processors

In order to evaluate the performance of our implementation we have conducted experiments using a variety of combinations among simulation input settings and experimental platform deployment. The experimental platform consisted of a host platform supplied with a quad-core Intel Xeon X5450 processor at the clock rate of 3.0 GHz with 4GB of main memory and one NVIDIA Tesla S1070 1U computing system. The Tesla S1070 system18 makes available 4 GPUs overall, with 30 stream multiprocessors and 240 cores on each GPU, resulting on an aggregate of 960 cores for our simulations. Each GPU is equipped with 4 GB of memory.

In the following figures we present results in terms of execution times (Fig. 6) and their respective speedups (Fig. 7 and Fig. 8) that were achieved in comparison with the sequential run. The computations refer to single precision arithmetic calculations and the presented results refer to the average of 10 simulation runs of the Rayleigh–Taylor instability were each simulation conducted 100 iterations. The results for the evaluation of oblique inviscid shock reflection are similar.

According to the experimentation results the highest performance is achieved when 4 GPUs are utilized. The succeeded speedup in that case is 53 on the average for the several mesh sizes. However this specific result is lower than the optimal if we compare it with the average speedup that we achieve when we utilize a single Tesla GPU. This is mainly due to the data transfers that have to take place between the host memory and the several device memories on each time step of the simulation. Nevertheless the speedup is still significant compared to the sequential or the OpenMP executions.

4 Conclusions

A high order WENO finite difference method was implemented for parallel processing with GPUs. A two level domain decomposition approach was employed in order to achieve highly accelerated processing. On the first level the domain was decomposed into subdomains that are assigned to multiple GPUs. On the second level within each GPU the domain was further decomposed into an appropriate number of thread blocks. High resolution simulations of oblique shock reflection and Rayleigh–Taylor instability were used as computational examples for the evaluation of the parallelization efficiency. It was found that a significant, however yet suboptimal, speedup was achieved with the available number of GPUs. It is anticipated that better handling of data transfer among GPUs can further increase processing speed to optimal levels. It is also expected that implementation of the existing parallelization approach on clusters with GPUs would make possible large scale DNS and LES computations.

References

[1] Garland, M., Le Grand, S., Nickolls, J., Anderson, J., Hardwick, J., Morton, S., Phillips, E., Zhang, Y., Volkov, V., "Parallel Computing Experiences with CUDA". *MICRO, IEEE,* Vol. 28, No. 4, 2008, pp. 13-27.

[2] Micikevicius, P. 2009. "3D Finite Difference Computation on GPUs using CUDA," In Proceedings of 2nd Workshop on General Purpose Processing on Graphics Processing Units (Washington, D.C., March 08 - 08, 2009). GPGPU-2, vol. 383. ACM, New York, NY, 79-84.

[3] Corrigan, A., Camelli, F., Lhner, R., and Wallin, J. "Running Unstructured Grid CFD Solvers on Modern Graphics Hardware," 19th AIAA Computational Fluid Dynamics Conference, July 2009, AIAA-2009-4001.

[4] Klckner, A., Warburton,T., Bridge, J., and Hesthaven, J. S., "Nodal discontinuous Galerkin methods on graphics processors," *J. Comput. Phys.* Vol. 228, 2009, pp. 7863-7882.

[5] PGI CUDA FORTRAN compiler. http://www.pgroup.com/resources/accel.htm

[6] Glimm, J., Grove, J., Li, X., Oh, W., and Tan, D.C, "The dynamics of bubble growth for Rayleigh–Taylor unstable interfaces," *Physics of Fluids,* Vol. 31, 1988, pp. 447–465.

[7] Young, Y.N, Tufo, H., Dubey, A., and Rosner, R., "On the miscible Rayleigh–Taylor instability: two and three dimensions," *Journal of Fluid Mechanics*, Vol. 447, 2001, pp. 377–408.

[8] Lindholm, E., Nickolls, J., Oberman, S., Montrym, J., NVIDIA Tesla: A Unified Graphics and Computing Architecture. Micro, IEEE 28, 2008, pp. 39-55.

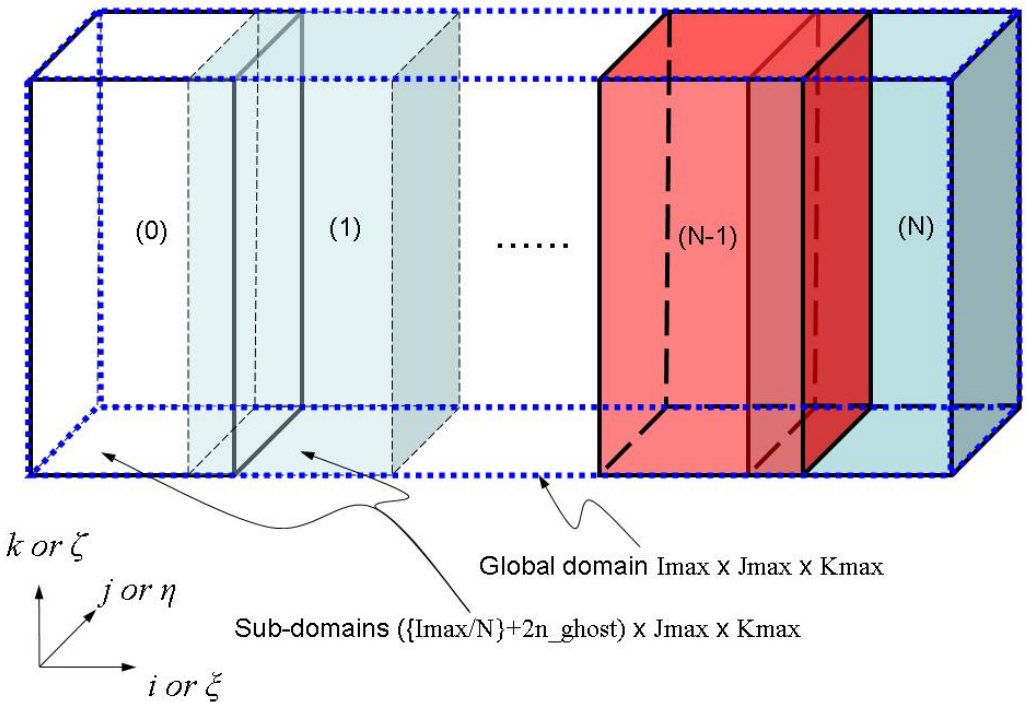


Figure 1. Schematic for domain decomposition with MPI or OpenMP

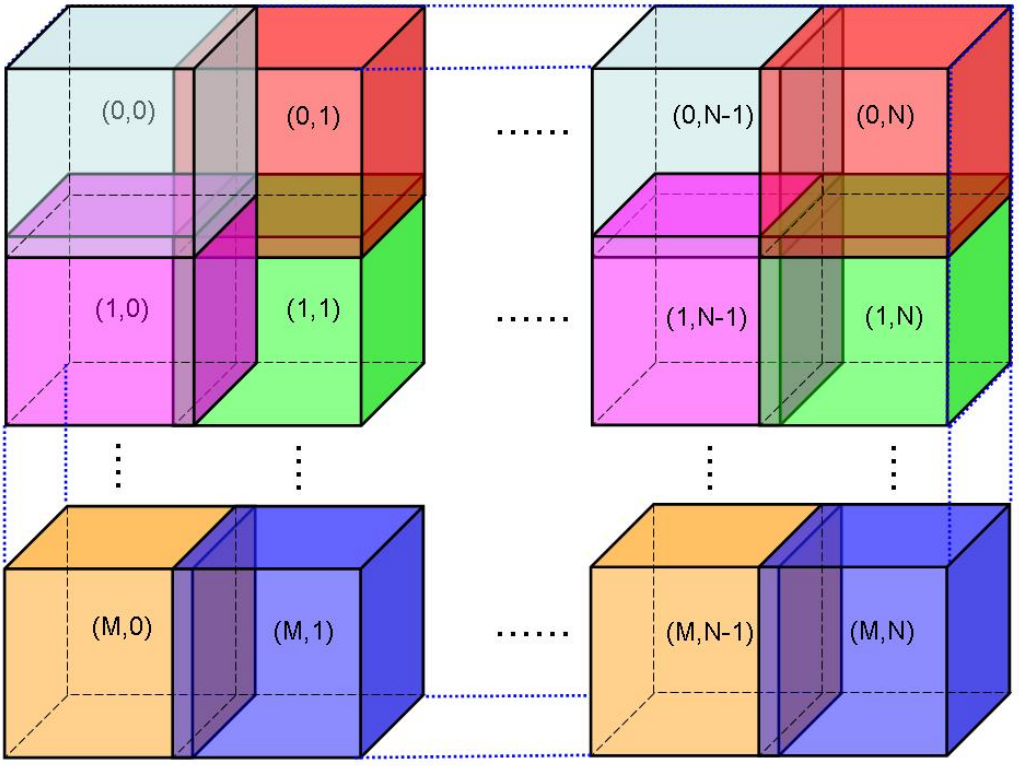


Figure 2. Schematic for domain decomposition in GPU parallelization

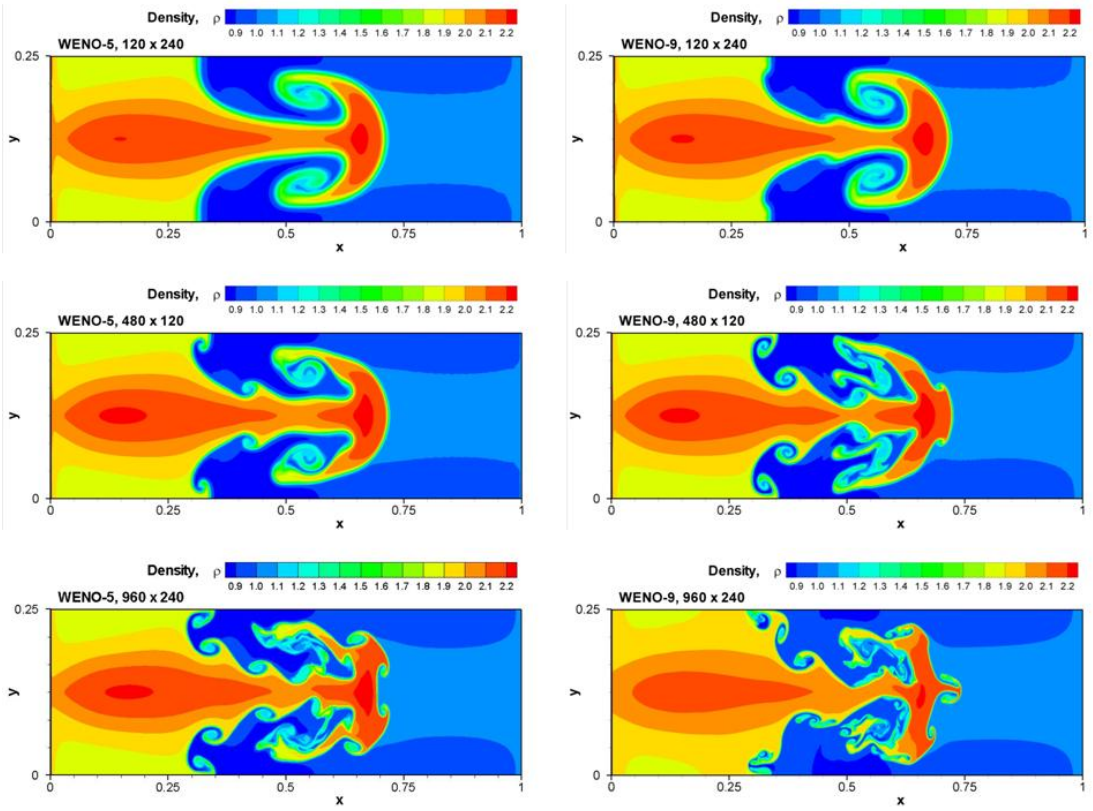


Figure 3. Effect of grid (h–type) refinement and order of accuracy (p – type) refinement on the resolution of the Rayleigh–Taylor instability

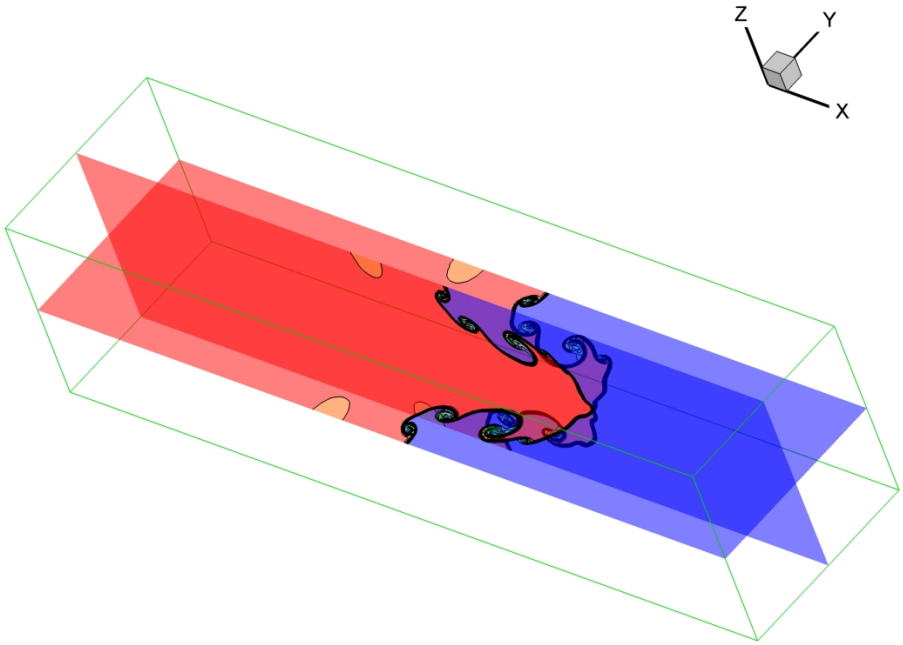


Figure 4. Numerical simulation of Rayleigh–Taylor instability using a 480 x 120 x 120 point mesh

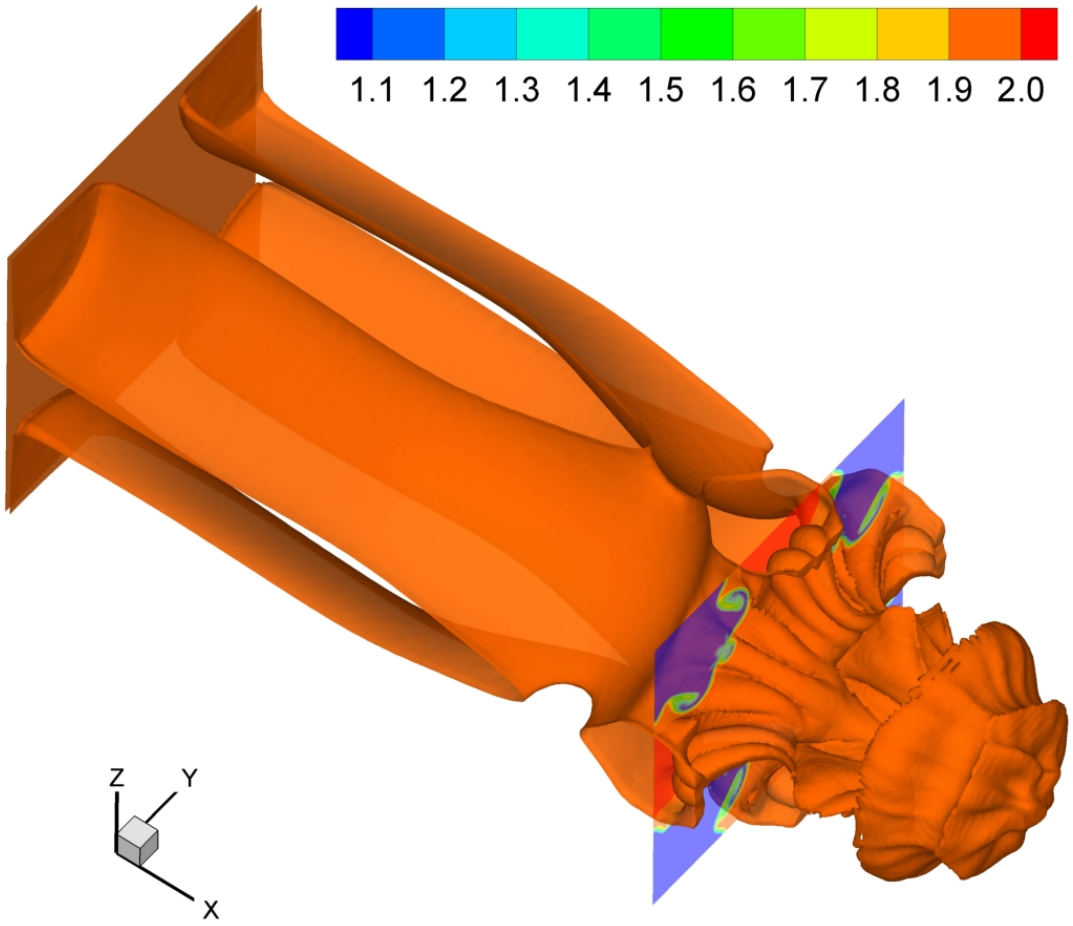


Figure 5. Numerical simulation of Rayleigh–Taylor instability using a 480 x 120 x 120 point mesh

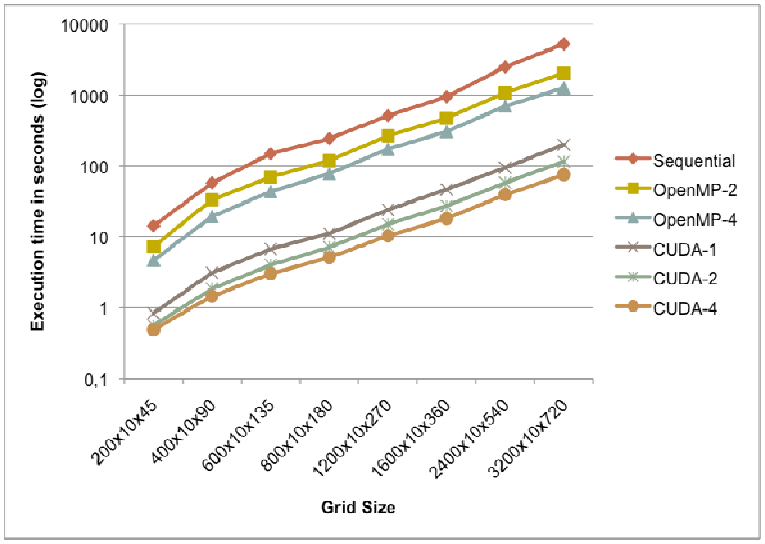


Figure 6. Execution times for several grid sizes

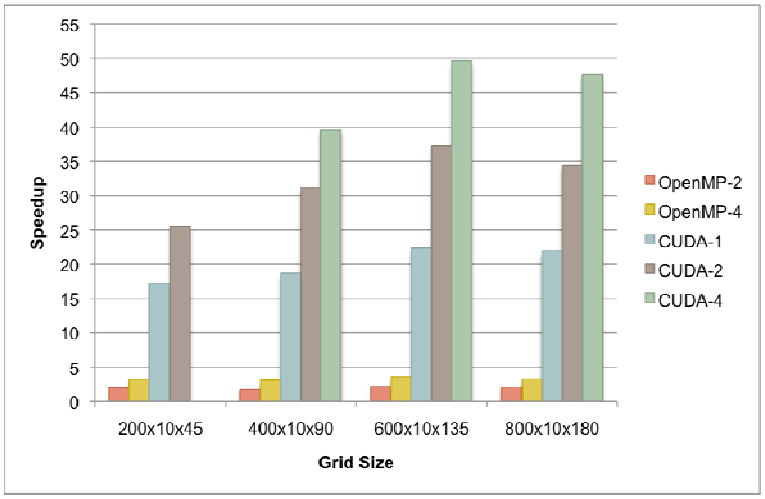


Figure 7. Speedup of OpenMP and GPU implementations compared to sequential execution I

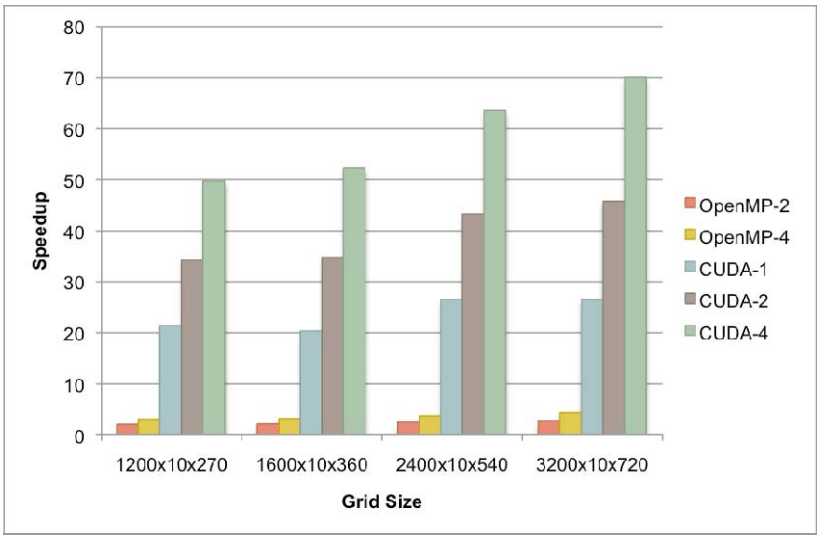


Figure 8. Speedup of OpenMP and GPU implementations compared to sequential execution II

Quarterly Progress Report for the
EOARD Project

Development of High–Order Methods for Multi–Physics Problems Governed by Hyperbolic Equations

by

John A. Ekaterinaris

FORTH / IACM

Award No. FA 8655–03–1–3085

Reporting period April 1 – June 30, 2010

A unified limiting approach for DG discretizations on mixed-type unstructured meshes

Summary

Accurate predictions of skin friction and thermal loads of high speed complex flows, in both simple and nontrivial geometries, require good shock capuring capability and high resolution. High order discontinuous Galerkin (DG) discretizations possess features that make them attractive for accurate computation of complex flows with strong shocks. A key ingredient that would make the DG method more attractive for these computations, is application of p-adaptive procedures that ensure accurate capturing of discontinuities with low order expansions and resolution of smooth complex features, such as vortices and shear layers, with higher order accuracy. In this work, a limiting procedure of DG discretizations is developed that is capable of computing high speed flows with strong shocks around complex geometries, using a p-adaptive procedure on mixed type (quadrilateral and triangular) meshes. The unified new approach for limiting high order expansion of the approximate solution for mixed type meshes is presented, and a number of results are shown to illustrate the potential of the method.

Introduction

The Discontinuous Galerkin (DG) method has become popular in recent years due to its ability to simulate flows around complex geometries and because it offers the advantage of increased order of approximation with a compact stencil. As a result, it yields high parallelization, both with domain decomposition and graphic processor units (GPUs). However, an aspect of the DG method that is not satisfactory as yet, is the ability to compute flows with discontinuities for arbitrary mixed type meshes and three dimensional computations. Usually, the DG method reduces the solution accuracy near discontinuities. This affects the resolution properties of the method and much of its potential is lost. Therefore, use of p-adaptive procedures is required to enhance utilization of the method for flows with discontinuities.

A TVB limiting procedure for the DG higher order expansions was introduced for scalar one dimensional hyperbolic conservation law by Shu [1] and Cockburn and Shu [2]. The extension of the method to one-dimensional systems has been applied with satisfactory results [3]. The basic idea behind was to use a flux limiter to preserve the monotonicity of the solution aver-

ages. The nonlinear limiting was applied in two dimensional problems [4–6], for second (P^1) and third (P^2) order accurate computations and for rectangular quatrilateral and triangular meshes. Qiu and Shu [7] applied the same TVB limiter for the one dimensional Euler equations, but reconstructed the solution at every element using a Hermite WENO scheme using the average solution at every element, and achieving a third order monotone solution. Similar approaches were followed by Zhu and Qiu [8, 9], and [10]. These limiting approaches are also restricted to meshes with rectangular elements and for triangular meshes with the use of an elaborate approach that is not straight forward to extend in three dimensions. The limiting approaches of [7–10], however, increase the stencil width and one of the main advantages of the DG method is lost. Furthermore, there is an ambiguity of the procedure regarding the solution's second derivative, and the case of distorted meshes has not been addressed.

Biswas [11] proposed an alternative implementation of limiting, where the solution moments are altered in order to preserve monotonicity. Krivodonova [12] applied also the same concept with promising results, in computations for flows with strong shocks and with order of accuracy higher than two. But, the meshes employed for these approaches did not have any distortion and used only rectangular elements.

The proposed approach uses the basic TVB limiter of Cockburn and Shu [2], and overcomes limitations associated with canonical quadrilateral meshes, it resolves the ambiguity concerning the second derivative of the solution and works very effectively on distorted mixed type meshes.

Unified local projection limiting

A spatial DG discretization of the Euler equations of gas dynamics is considered. A P^1 approximation of the solution is applied, using modal bases, which are constructed by the tensor product of *Legendre* polynomials [13] on the standard square reference element of Fig. 1. Arbitrary quadrilaterals of the physical space (x, y) transform to the standard square element in the reference space (n_1, n_2). Triangles in the physical space (see Fig. 2) transform to right triangles in the (ξ_1, ξ_2) space and then to the standard square in the (n_1, n_2) space, through the collapsed coordinate transformation [13]. Details of the quadrilateral ans triangle transformations are shown in Figs. 1 and 2, respectively.

The current approach preserves the monotonicity of the solution in a TVB sense, by applying a slope limiting procedure, which is transparent to the element type. Specifically, all the limiting operations are performed for every

element in the reference space (n_1, n_2) for the standard square configuration, where it is required that the slope of the solution at the element edges does not exceed the variation of the mean solution across them.

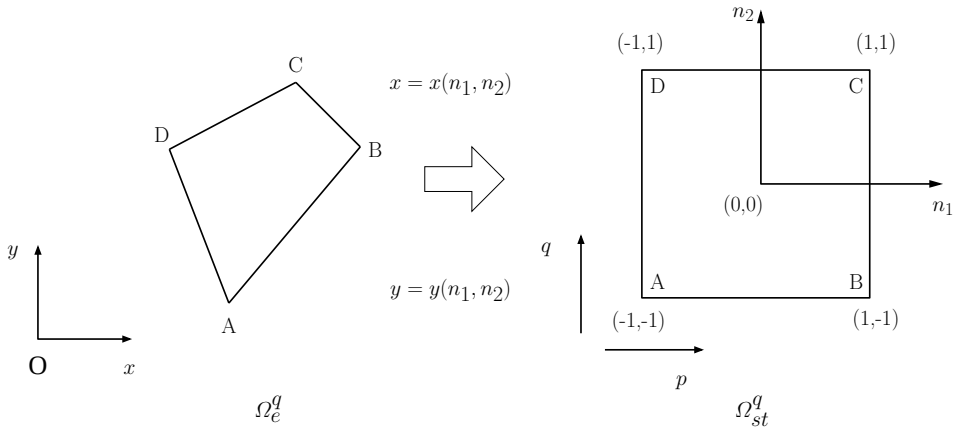


Figure 1: Transformation of the physical quadrilateral (x, y) to the standard rectangular element configuration $(n_1, n_2 \in [-1, 1])$.

In two dimensions, every element is checked for limiting in each direction n_1, n_2 of the reference space by using the TVB limiter proposed by Shu [1]:

$$\overline{m}(a_1, a_2, \cdots, a_n) = \begin{cases} a_1 & \text{if } |a_1| \leq ML_{x,y}^2, \\ m(a_1, a_2, \cdots, a_n) & \text{otherwise,} \end{cases} \tag{1}$$

where $M > 0$ is a parameter that estimates the second derivative of the solution, and $L_{x,y}$ a characteristic local mesh length in the x and y direction at every element. The function $m(a_1, a_2, \cdots, a_n)$, is the usual minmod function:

$$m(a_1, \cdots, a_n) \begin{cases} s \cdot min_{1 \leq j \leq n} |a_j| & \text{if } sign(a_1) = \cdots = sign(a_n) = s, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The limiting is performed in the characteristic space, by computing the directional Jacobian using the Roe averaged variables. An element is "flagged" for limiting, if the limiter in Eq. (1) returns an argument other than the first.

For the effective use of the limiter, it is crucial to have a good estimate of the second order derivative of the solution and of the local mesh lengths. The characteristic local lengths are computed as in [15]:

$$L_x = \frac{1}{S} \sum_e f_{x,e} \Delta y_e, \quad L_y = -\frac{1}{S} \sum_e f_{y,e} \Delta x_e, \tag{3}$$

where f_x, f_y are the following grid functions:

$$f_x = \Delta x_c^e |\Delta x_c^e|, \quad f_y = \Delta y_c^e |\Delta y_c^e|. \tag{4}$$

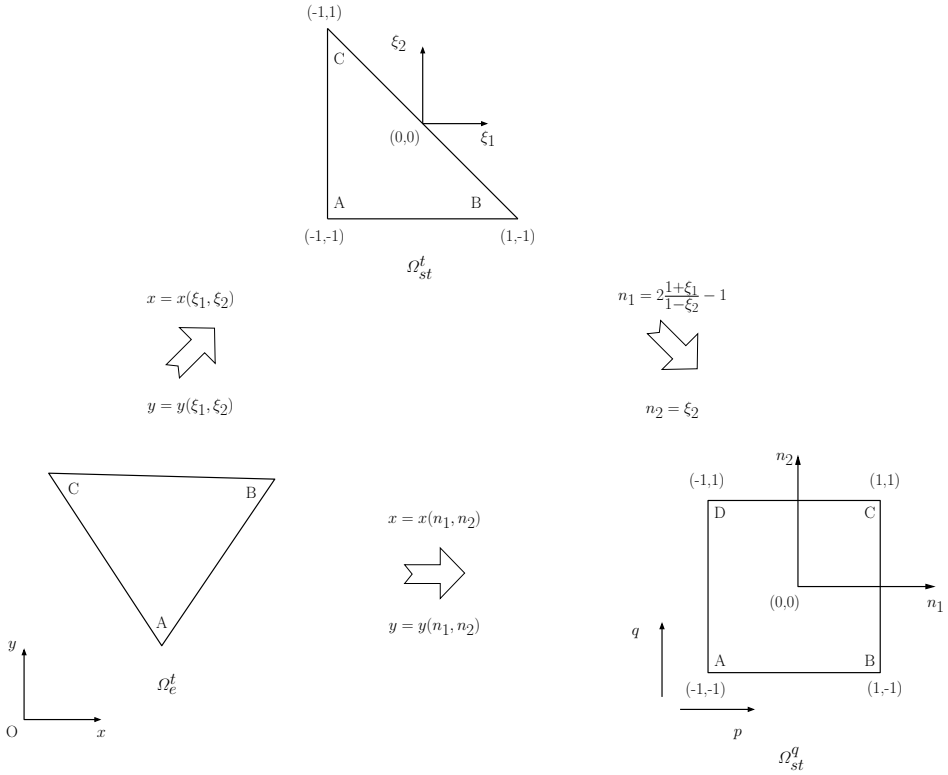


Figure 2: Transformation of the physical triangle to the standard triangular element ($\xi_1, \xi_2 \in [-1, 1]$ with $\xi_1 + \xi_2 \leq 1$) and standard quadrilateral element configuration ($n_1, n_2 \in [-1, 1]$).

The parameter M in our approach is not chosen arbitrarily as in Shu [3], where a constand value of M=50 is used, but is set equal to the absolute value of the estimate of the *Laplacian* operator for the element k where limiting is applied:

$$M = \left| \sum_e (\overline{u}_e - \overline{u}_k) \right|. \tag{5}$$

where with $\overline{u}$ is denoted the mean value of the solution, and the index e counts all the elements sharing the same edges with element k.

Limiting Quadrilateral elements

The mapping of a quadrilateral element to the computational space is shown in Fig. 1 1. For a second order accurate solution, P^1 approximation, the

following expansion of the solution is used:

$$u(\mathbf{x},t) = c_0(t)b_0(\mathbf{x}) + c_1(t)b_1(\mathbf{x}) + c_2(t)b_2(\mathbf{x}) + c_3(t)b_3\mathbf{x}, \tag{6}$$

where $b_i(\mathbf{x}), c_i(t),\ i = 0, 1, ..., 3$ are the expansion bases and the coefficients of the expansion, respectively. For the standard square element, the modal bases with respect to the local Cartesian system (n_1, n_2), of the reference element are:

$$\begin{aligned} b_0 &= 1, \\ b_1 &= n_1, \\ b_2 &= n_2, \\ b_3 &= n_1 n_2. \end{aligned}$$

The expansion coefficients are the solution moments, and correspond to the derivatives u_x, u_y and u_{xy}, respectively. In order to keep the solution monotone, as much as possible, it is imperative to limit the coefficients corresponding to the P^1 part of the solution. The midpoints of the edges are chosen for limiting of the coefficients, for the following reasons:

- Avoid ambiguity and closure problems.
- Computational efficiency

Along the n_1 direction coefficient $c_1 \sim u_x$ is limited. For doing so, the P^1 part of the solution at the midpoints of the edges BC and AD (see Fig. 3) is used. According to Eq. (6):

$$U_{BC} = u(1, 0) - c_0 = c_1, \tag{7a}$$

$$U_{AD} = u(-1, 0) - c_0 = -c_1, \tag{7b}$$

and the value for the coefficient c_1 is:

$$c_1 = \frac{U_{BC} - U_{AD}}{2}. \tag{8}$$

Limiting of the solution along the n_1 direction is performed by limiting the values U_{BC} and U_{AD} at the midpoints of the edges as follows:

$$\widetilde{U}_{BC,(i,j)} = m(U_{BC,(i,j)}, c_{0,(i+1,j)} - c_{0,(i,j)}, c_{0,(i,j)} - c_{0,(i-1,j)}), \tag{9a}$$

$$\widetilde{U}_{AD,(i,j)} = m(U_{AD,(i,j)}, c_{0,(i,j+1)} - c_{0,(i,j)}, c_{0,(i,j)} - c_{0,(i,j-1)}), \tag{9b}$$

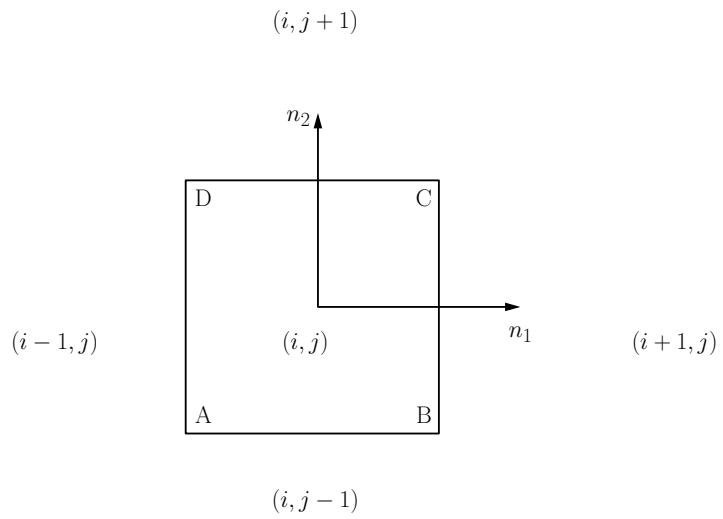


Figure 3: Stencil for limiting over the standard square element.

and the new value of c_1 is computed by Eq. (8). Similar methodology is followed for the $c_2 \sim u_y$ coefficient.

If a quadrilateral element is flagged for limiting, then the coefficient $c_3 \sim u_{xy}$ is set to zero, and moreover, the residual associated with the corresponding base is set to zero, during a multi-step time marching method. The rationale behind this is that, as long as, the coefficient c_3 is set to zero, it must be kept zero along the stages between time steps.

Limiting Triangular elements

In Fig. 2, the mapping of a triangular element in the reference computational domain is given. Applying a P^1 approximation over the elemental domain, the following expansion of the solution is used:

$$u(\mathbf{x}, t) = c_0(t)b_0(\mathbf{x}) + c_1(t)b_1(\mathbf{x}) + c_2(t)b_2(\mathbf{x}). \tag{10}$$

Referring to Fig. 2, the modal bases relative to the Cartesian system (n_1, n_2) of the reference element are:

$$\begin{aligned} b_0 &= 1, \\ b_1 &= n_1 \frac{1 - n_2}{2}, \\ b_2 &= \frac{3n_2 + 1}{2}. \end{aligned}$$

The coefficients c_1 and c_2 correspond to u_x and u_y, respectively. The midpoints of the edges are chosen for the limiting procedure, and a similar procedure as in the quadrilaterals is followed. Specifically, along the n_1 direction, coefficient c_1 is altered, by limiting the P^1 part of the solution. Then, from the expansion given in Eq. (10), coefficient c_1 is computed as:

$$c_1 = U_{BC} - U_{AD}. \tag{11}$$

Along the n_2 direction coefficient c_2 is limited, but as a collapsed Cartesian system is used for representing the bases over the standard square element, the P^1 part of the solution across the edge AB in Fig. 3 is limited only with the mean solution variation across that edge. The final value of the c_2 coefficient is:

$$c_2 = m(U_{AB,(i,j)}, c_{0,(i,j)} - c_{0,(i,j-1)}, c_{0,(i,j)} - c_{0,(i,j-1)}). \tag{12}$$

Results

A third order explicit TVD Runge-Kutta scheme [14] is used for time marching. The new limiting procedure is applied for several cases with triangular and quadrilateral meshes with arbitrary (not rectangular) shape. Some examples are shown next.

Oblique shock reflection

The problem of an oblique shock at Mach 3 is solved. The domain is a 4x1 rectangle. On the top, the flow is specified with the oblique shock relations for a shock at M = 3 and an angle $\theta = 30°$. At the inflow uniform supersonic flow is specified. On the wall $y = 0$ the slip velocity condition is applied, while density and pressure are extrapolated assuming zero normal gradient. At the supersonic outflowm all quantities are extrapolated from the interior. A mesh of 200×50 triangular elements has been used. The pressure contours are shown in Figs. 4a,4b. Good capturing of both the incident and the reflected shocks is achieved. In Fig. 5, the line plot of the normalized pressure along the center of the domain $y = \frac{1}{2}$ is shown.

Supersonic flow over a cylinder

Supersonic flow at a free stream Mach number $M = 2$ over a cylinder is considered next. Both structured and unstructured meshes have been used. The meshes used in the computations of the flow are shown in Fig 6, and they

consist of 2400 and 4800 elements of quadrilaterals and triangles, respectively. Due to the symmetry of the problem only half of the domain is considered.

In Fig. 7, the pressure contours are shown. The agreement between the numerical solutions computed on different meshes appears satisfactory. Computed pressure and velocity magnitude along the stagnation line are compared in Figs. 8 and 9, respectively. Good quantitative agreement between the computed results is obtained.

Conclusions

A new limiting procedure for DG discretizations has been developed and applied for distorted meshes with arbitrary quadrilaterals and triangular meshes. The proposed approach applies to mixed-type meshes and it is straight forward to extend in three dimensions. Numerical solutions for triangular and quadrilateral meshes were presented. Numerical solutions of viscous flows with shocks will be computed using a p-adaptive procedure.

Bibliography

[1] C.-W. Shu, TVB uniformly high-order schemes for conservation laws, Mathematics of Computation, 49, 105, 1987.

[2] B. Cockburn, and C.-W. Shu, TVB Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws II: General framework, Mathematics of computation, 52, 411, 1989.

[3] B. Cockburn, S.-Y. Lin and C.-W. Shu, TVD Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws III: One-dimensional Systems, Journal of Computational Physics, 84, 90, 1989.

[4] B. Cockburn, S. Hou and C.-W. Shu, The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multi-dimensional case, Journal of Computational Physics, 54, 545, 1990.

[5] B. Cockburn, and C.-W. Shu, P^1-RKDG method for two-dimensional Euler equations of gas dynamics, In Proc. Fourth Int. Symp. on CFD, UC Davis, 1991.

[6] A. Burbeau, P. Sagaut and Ch.-H. Bruneau , A Problem-Independent Limiter for High-Order RungeKutta Discontinuous Galerkin Methods, Journal of Computational Physics, 169, 111, 2001.

[7] J. Qiu and C.-W. Shu, Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galerkin method: one dimensional case, Journal of Computational Physics, 193, 115, 2003.

[8] J. Zhu and J. Qiu, Local DG method using WENO type limiters for convectiondiffusion problems Journal of Computational Physics, In Press, Corrected Proof, Available online 21 March 2010.

[9] J. Zhu, J. Qiu, C.-W. Sh and M. Dumbser, RungeKutta discontinuous Galerkin method using WENO limiters II: Unstructured meshes, Journal of Computational Physics, 227, 4330, 2008.

[10] H. Luo, J. D. Baum, R. Lhner, Hermite WENO-based limiter for discontinuous Galerkin method on unstructured grids Journal of Computational Physics, 225, 686, 2007.

[11] R. Biswas, K. D. Devine, J. E. Flaherty, Parallel, adaptive finite element methods for conservation laws, Applied Numerical Mathematics, 14, 255, 1994.

[12] L. Krivodonova, Limiters for high-order discontinuous Galerkin methods, Journal of Computational Physics, 226, 879, 2007.

[13] G. E. Karniadakis, S. J. Sherwin, Spectral/*hp* Element methods for CFD, 2nd edition, Oxford University Press, New York, 2003.

[14] B. Cockburn, G. E. Karniadakis and C.-W. Shu, The development of discontinuous Galerkin method, in: B. Cockburn, G. E. Karniadakis and C.-W. Shu (Eds.), Discontinuous Galerkin Methods, Theory, Computation and Applications, Lecture Notes in Computational Science and Engineering, 11, Springer, New York, 2000.

[15] I. Kallinderis, C. Kontzialis, A priori mesh quality estimation via direct relation between truncation error and mesh distortion, Journal of Computational Physics, 228, 881, 2009

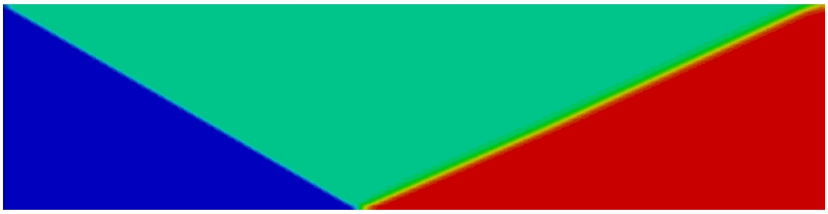


Figure 4a: Pressure contours for the oblique shock problem.

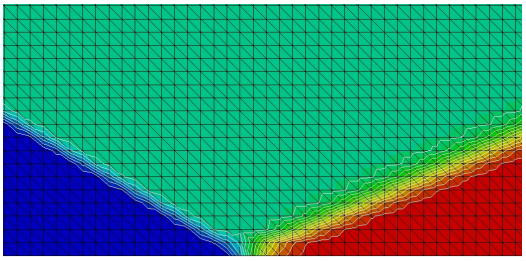


Figure 4b: Pressure contours for the oblique shock problem (Detail).

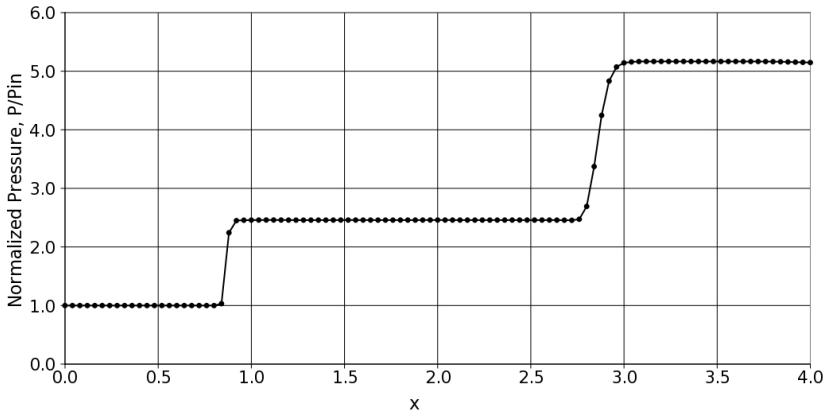
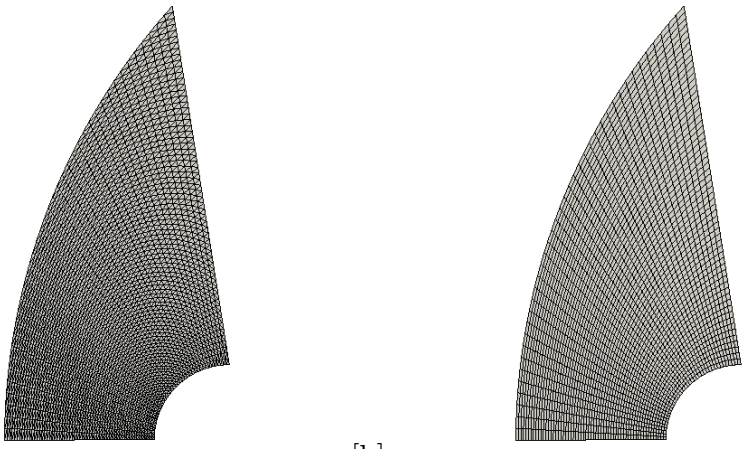
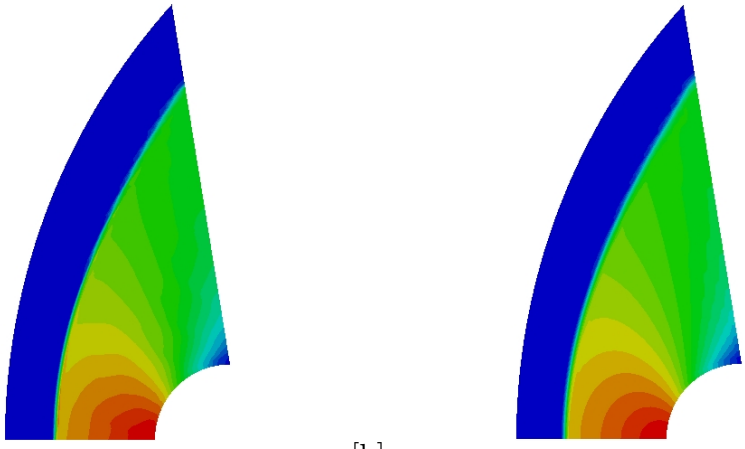


Figure 5: Pressure plot at $y = \frac{1}{2}$ for the oblique shock problem.



[a] [b]

Figure 6: Meshes used of a Mach 2 flow over a cylinder: (a) Triangles, (b) Quadrilaterals.



[a] [b]

Figure 7: Pressure contours for a Mach 2 flow over a cylinder: (a)Triangles , (b) Quadrilaterals.

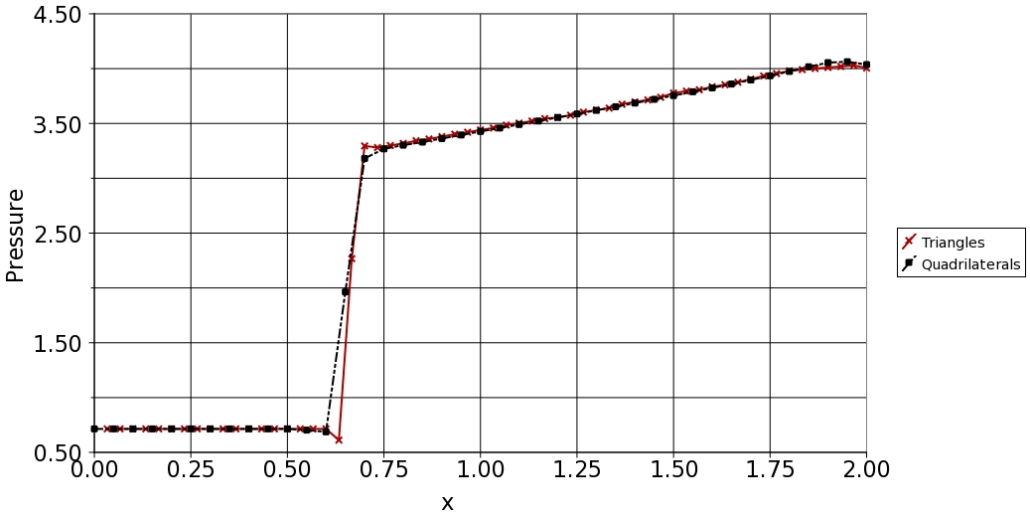


Figure 8: Pressure plot along the stagnation line.

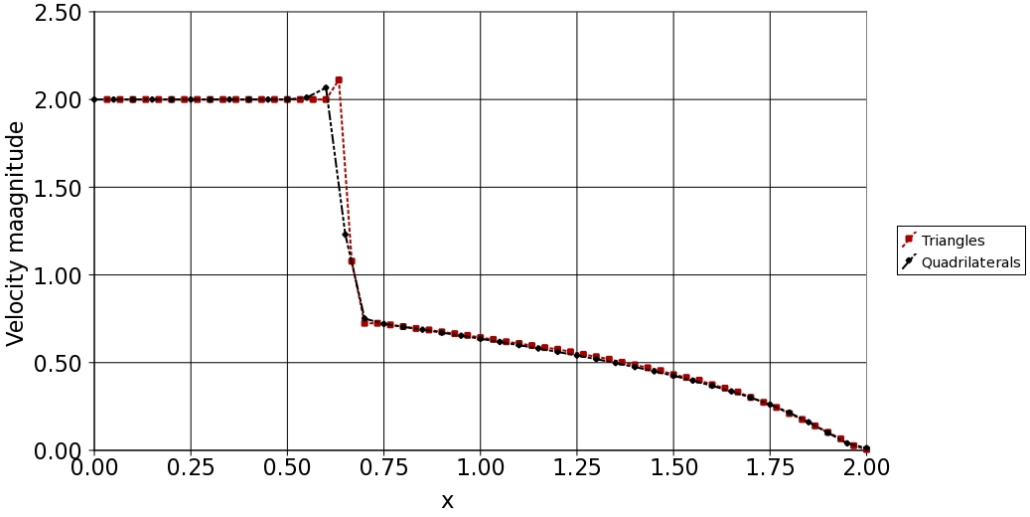


Figure 9: Velocity magnitude along the stagnation line.

Quarterly Progress Report for the
EOARD Project

Development of High–Order Methods for Multi–Physics Problems Governed by Hyperbolic Equations

by

John A. Ekaterinaris

FORTH / IACM

Award No. FA 8655–07–1–3099

Reporting period July 1 – September 30, 2010

Parallelization of the DG method for the Navier-Stokes Equations

Summary

Accurate predictions of skin friction and thermal loads of high speed complex flows, in both simple and nontrivial geometries, require good shock capturing capability and high accuracy in the viscous flow region. High order discontinuous Galerkin (DG) discretizations possess features that make them attractive for accurate computation of complex flows with strong shocks. A key ingredient that would make the DG method more attractive for these computations, is application of p-adaptive procedures that ensure accurate capturing of discontinuities with low order expansions and resolution of smooth complex features, such as vortices and shear layers, with higher order accuracy. A limiting procedure of DG discretizations capable of commuting high speed flows with strong shocks around complex geometries, using a p-adaptive procedure on mixed type (quadrilateral and triangular) meshes was developed and preliminary results were presented in the previous report. In this report further validation of the limiting approach is presented for standard computationally demanding flow cases, such as the Mach reflection and the wind tunnel with a step. Application examples for both quadrilateral, triangular and mixed type meshes are shown.

These large scale computations were made possible by parallelizing the code using domain decomposition and MPI. Viscous terms were added for the DG method using the local DG approach. Incorporation of an implicit time marching scheme, which will make possible high Reynolds number computations, is underway for a single processor. Implementation of implicit time marching with domain decomposition is more involved and requires use of a dual time stepping for time accurate computations. Developments of capabilities to compute viscous flow numerical solutions with implicit time stepping for multiprocessor will be presented in the next report.

Parallelization

The compact form of the DG discretization stencil (information is needed only from the immediate neighbors) permits efficient implementation on parallel processors. The data structure of the code was designed in order to meet the requirements of the DG method on mixed type element meshes and p-adaptive expansions. Moreover, it allows application of the domain decomposition method through MPI without serious additional coding complexity.

Every simulation with domain decompositions initiates with the partitioning of the computational domain using free available software Metis (http://glaros.dtc.umn.edu/gkhome/views/metis). In order to handle mixed type meshes, a graph file representing the mesh is initially created and subsequently partitioning of the graph in to subdomains leads to the final decomposition of the mesh. Every partition (subdomain) includes layers of elements belonging to neighboring partitions. This set of elements is the receiving list for a partition, and the set of elements of the partition sharing the same edges with the receiving list, is the sending list to neighboring partitions.

A TVD Runge–Kutta method was used for time marching. Limiting is applied after every stage of the RK cycle. Finally, an exchange between the partitions of the solution is also performed at the end of each stage. For a parallel p-adaptive DG code, all coefficients of the expansion for the highest order polynomial approximation are exchanged between partitions of the mesh.

The parallel algorithm for every partition is as follows:

1. Transfer the solution (expansion coefficients) at the end of the RK cycle between partitions. Wait for the transfer to complete.
2. Apply the limiter.
3. Transfer of all coefficients of the highest polynomial application. Wait for the transfer to complete.
4. Assign solution coefficients from P0 up to the highest polynomial application.
5. Compute the volume integrals.
6. Compute the line integrals.
7. Compute residual for every element belonging to the partition.
8. Enter the next RK stage.
9. Complete the RK cycle and repeat.

Results

Results from parallel computations of flows with strong shocks on single- and mixed-type meshes are presented next. The supersonic flow over a cylinder is the first example. The single type computational mesh with triangular elements and the computed density field for flow at M=2.0 is are shown in Figs. 1-2. The same flow was computed on a mesh with arbitrary quadrilateral elements (see Figs. 3 and 4). This computation was not possible with the original limiting procedure suggested by Cockburn and Shu [1] and difficult to implement with limiting approaches presented in Ref. 2-6, but it is straightforward with our new limiting approach. Next, an example of a computation on a mixed-type mesh is presented. The mesh and the computed density field is shown in Fig. 5. For this computation, a p-adaptive procedure is applied where the flow near the discontinuity is computed with P1 polynomials as flagged by the limiter while the rest of the smooth flow field is computed with P2 polynomials. A comparison of the computed density along the symmetry line is shown in Fig. 6. The agreement with the quadrilateral mesh solution is excellent.

Computations for two classical examples are presented next. These are the M=3.0 tunnel with a step and the double Mach reflection of a strong shock at Mach=10 (P. Woodward and P. Colella, "The numerical simulation of two-dimensional fluid flow with strong shocks," Journal of Computational Physics, Vol. 54, No. 1, 1984, pp. 115-173). The computational domain for the tunnel with a step discretized with a mixed type mesh and the partition of this mesh to subdomains for parallel processing is shown in Fig. 7. At the corner the mesh was refined in order to diminish the Mach stem that is created from the artificial entropy layers caused by the sharp corner. The computed flow field and the elements flagged for limiting are shown in Fig. 8. The elements flagged for limiting for the double Mach reflection problem computation are shown in Fig. 9 and the computed flow field is shown in Fig. 10. For these computations the elements flagged for limiting are updated continuously during the time accurate computation. Computations with higher order expansions using p-adaptive procedures are underway and they will presented at the AIAA ASM meeting in January 2011.

Conclusions and Outlook

The DG code was parallelized using domain decomposition and MPI. The new limiting procedure for DG discretizations was then applied for standard computationally demanding test problems. The proposed approach applied

to quadrilateral, triangular, and mixed-type meshes. The extension of the limiting approach in three dimensions is underway. Numerical solutions of viscous flows with shocks will be computed using a p-adaptive procedure. For high Reynolds number flows and long time integration with small size meshes use of an implicit time marching scheme is necessary.

Bibliography

[1] B. Cockburn, G. E. Karniadakis and C.-W. Shu, The development of discontinuous Galerkin method, in: B. Cockburn, G. E. Karniadakis and C.-W. Shu (Eds.), Discontinuous Galerkin Methods, Theory, Computation and Applications, Lecture Notes in Computational Science and Engineering, 11, Springer, New York, 2000.

[2] A. Burbeau, P. Sagaut and Ch.-H. Bruneau , A Problem-Independent Limiter for High-Order RungeKutta Discontinuous Galerkin Methods, Journal of Computational Physics, 169, 111, 2001.

[3] J. Qiu and C.-W. Shu, Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galerkin method: one dimensional case, Journal of Computational Physics, 193, 115, 2003.

[4] J. Zhu and J. Qiu, Local DG method using WENO type limiters for convection-diffusion problems Journal of Computational Physics, In Press, Corrected Proof, Available online 21 March 2010.

[5] J. Zhu, J. Qiu, C.-W. Sh and M. Dumbser, Runge-Kutta discontinuous Galerkin method using WENO limiters II: Unstructured meshes, Journal of Computational Physics, 227, 4330, 2008.

[6] H. Luo, J. D. Baum, R. Lohner, Hermite WENO-based limiter for discontinuous Galerkin method on unstructured grids Journal of Computational Physics, 225, 686, 2007.

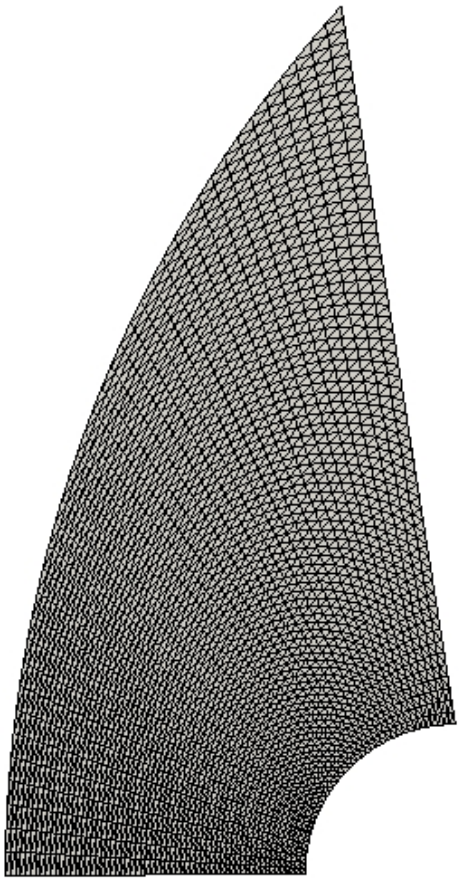


Figure 1. Triangular element mesh over the cylinder

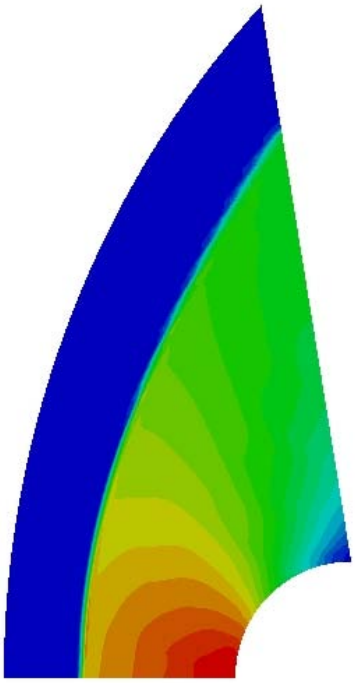


Figure 2. Computed density field at M=2.

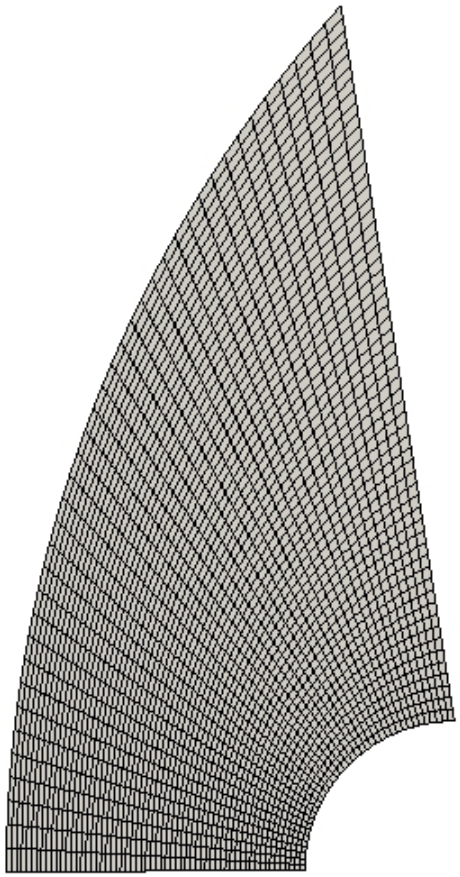


Figure 3. Quadrilateral element mesh over the cylinder

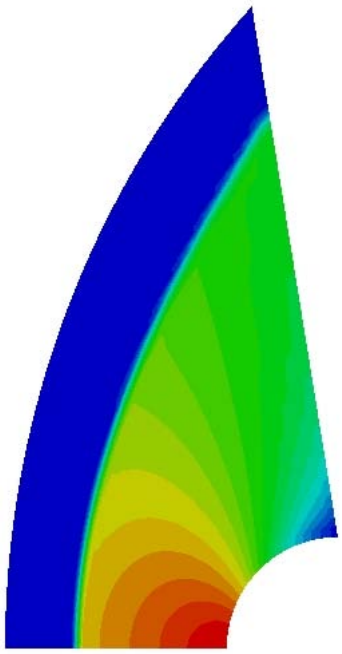


Figure 4. Computed density field at M=2.

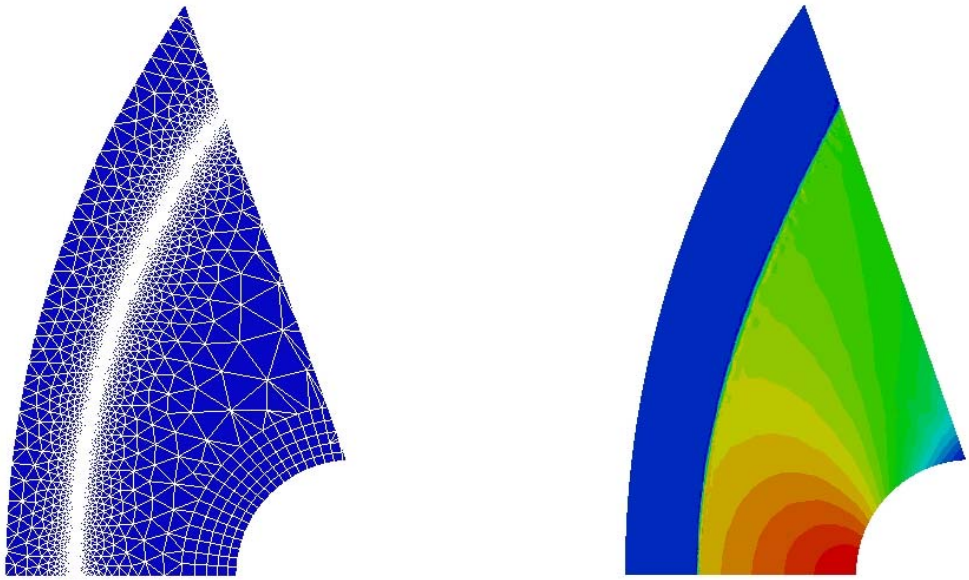


Figure 5. Mixed-type mesh for the computation of supersonic flow.

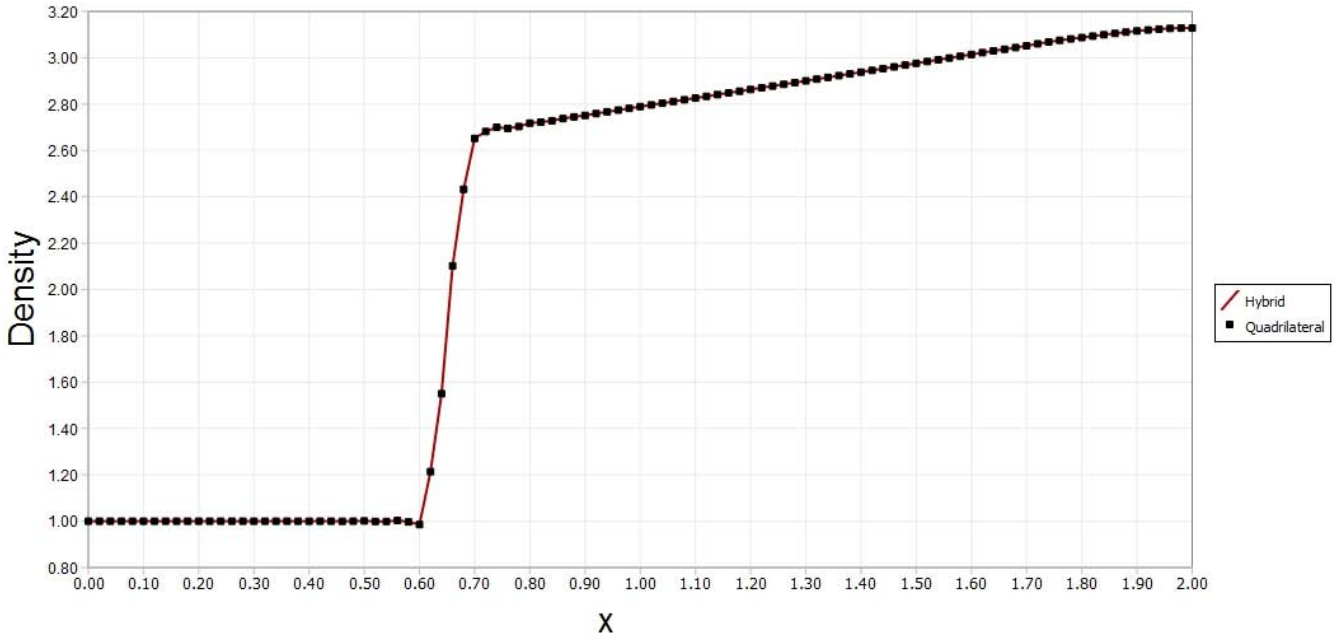


Figure 6 Comparison of the density along the symmetry stagnation line obtained with quadrilateral (P1) and mixed type meshes (p2 adaptive).

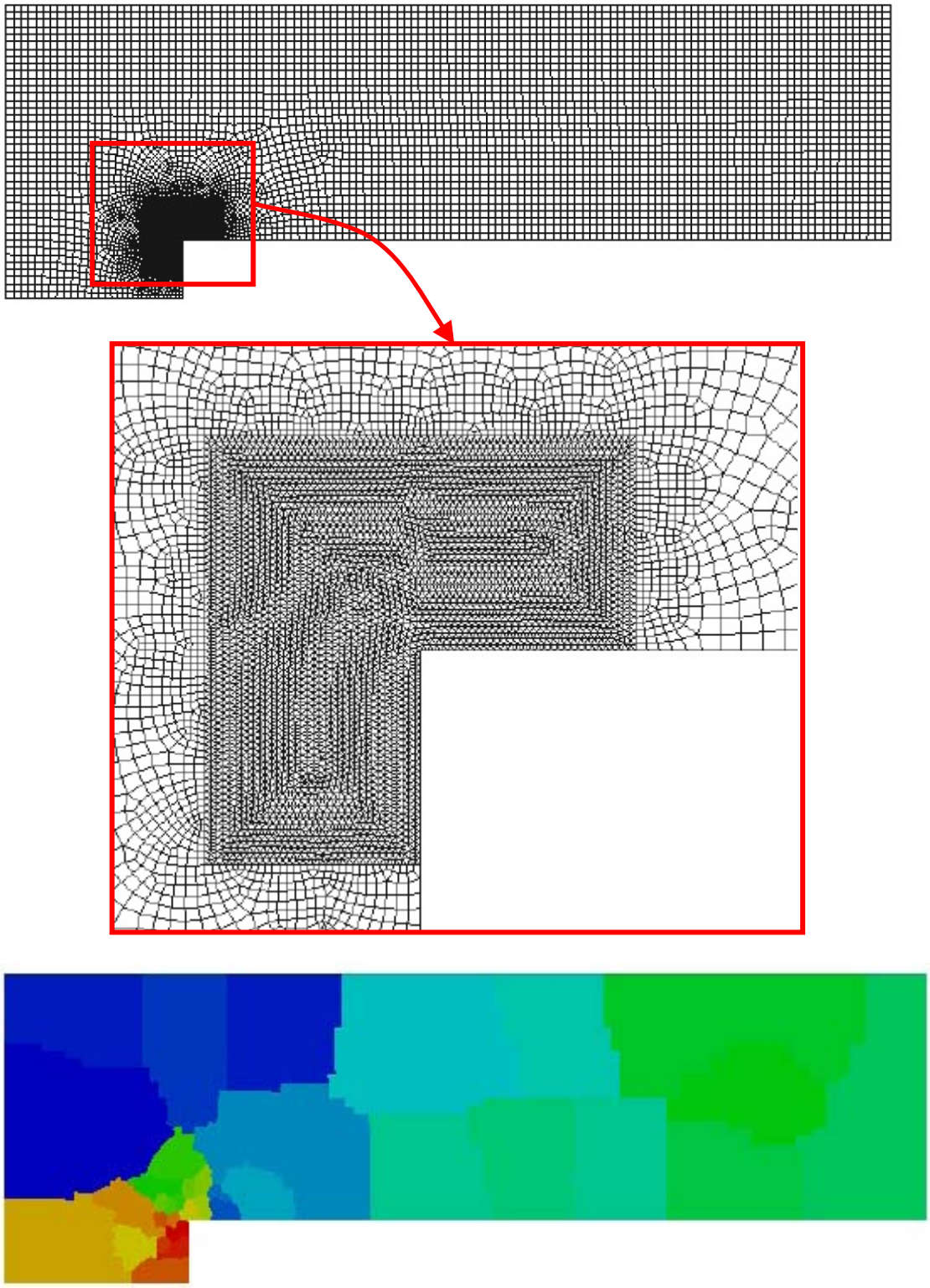
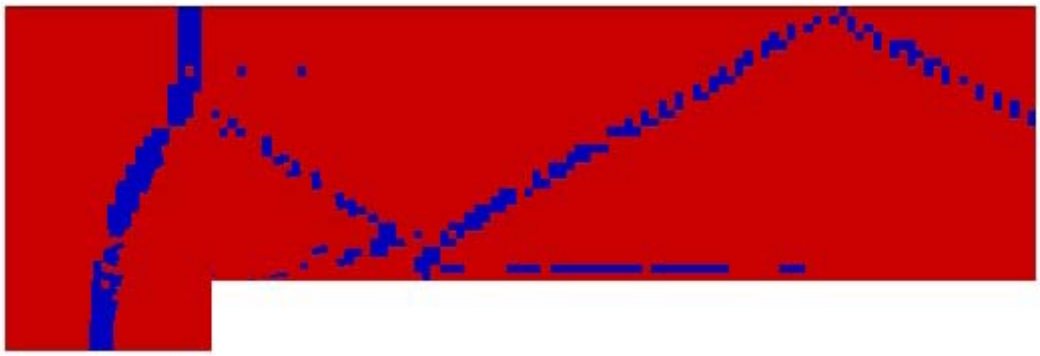
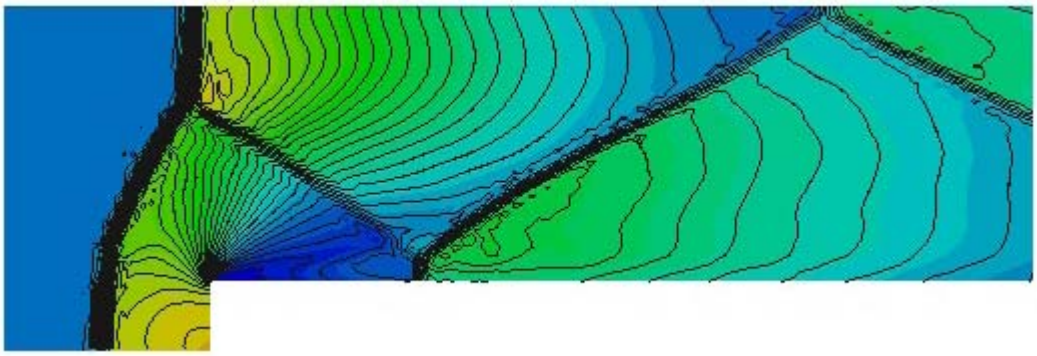


Figure 7. Mixed-type mesh and partition for MPI parallel processing.



(elements flagged for limiting)



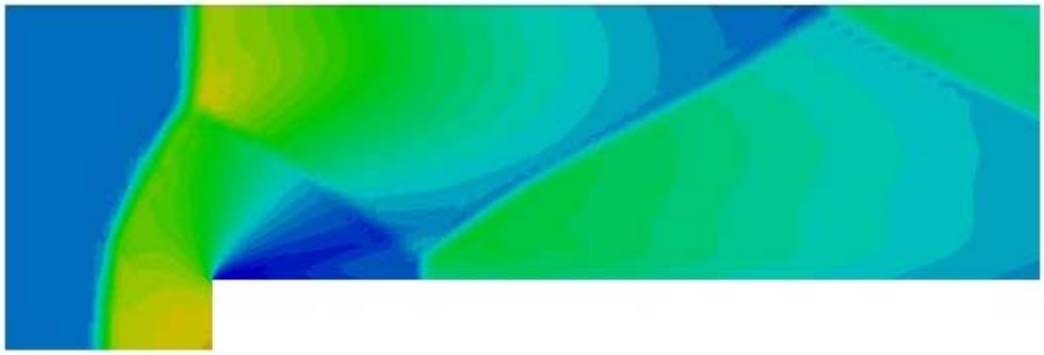


Figure 8. Elements flagged for limiting (top) and computed density and pressure for the flow in a tunnel with a step.

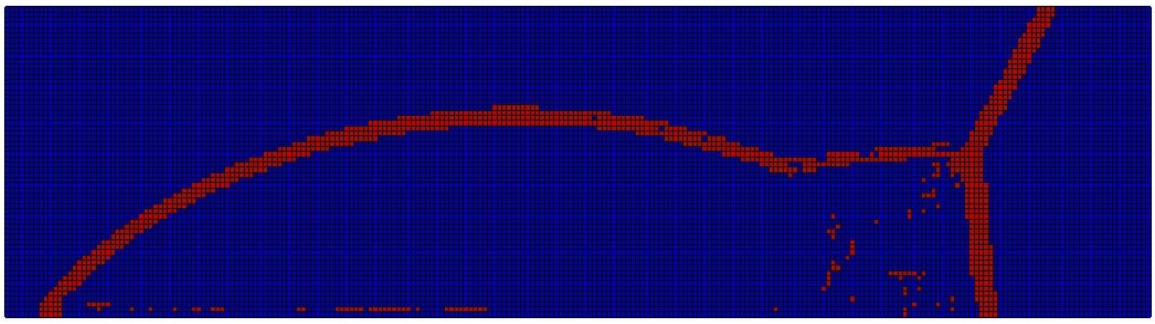


Figure 9. Elements flagged for limiting

Figure 11. Computed density field.