TOWARDS AN EFFICIENT AND ROBUST HIGH ORDER ACCURATE FLOW SOLVER FOR VISCOUS COMPRESSIBLE FLOWS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF AERONAUTICS AND ASTRONAUTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Sachin Premasuthan
March 2010

Abstract

The field of high-order methods for unstructured grids has seen a surge of research activity over the last decade, particularly due to its potential in applications such as Large Eddy Simulation (LES), Direct Numerical Simulation (DNS), Computational Aero-Acoustics (CAA), turbulent combustion and vortex dominated flows. However, this field has not yet reached the level of maturity required to solve real flow problems on complicated geometries. The present thesis work makes an effort towards the realization of a flow solver that is high-order accurate but also efficient, robust and geometrically flexible at the same time. The approach to high-order spatial discretization is based on the Spectral Difference (SD) method for unstructured quadrilateral meshes. High-order (quadratic and cubic) representation is used for curved boundary surfaces.

Due to their slow convergence rates with standard explicit time-stepping schemes, the use of high order schemes becomes viable only when there is some kind of convergence acceleration technique used. Towards this end, the multi-order (p-multigrid) method and Implicit time-stepping are implemented, and significant convergence acceleration is achieved for both steady and unsteady flow problems.

Another significant challenge with using high-order methods is their inability to handle flow discontinuities. An artificial viscosity based approach is designed and implemented to enable computation of flows with shocks. Promising results have been obtained, with the proposed method being able to produce stable solutions with sharp resolution of shocks, and no significant spurious oscillations.

Adaptive mesh and order refinement have great potential in reducing the computational effort required to reach a specified level of accuracy, particularly in the context

of high-order formulations. The capability for adaptive mesh and order refinement is enabled using mortar elements to handle non-conforming solution approximations at the cell interfaces. Adaptive mesh refinement has been combined with artificial viscosity to enhance the resolution of discontinuities and solution accuracy.

As part of this work, a 2D flow solver is developed, tested, validated and applied to a variety of flow problems. The current work also demonstrates the effectiveness of computational tools such as convergence acceleration, shock-capturing and adaptive refinement, for enhancing the efficiency and robustness of high order flow computations on unstructured grids.

Acknowledgements

First and foremost, I would like to gratefully acknowledge my advisor, Professor Antony Jameson, for giving me an opportunity to complete my PhD under his guidance. He has been a great mentor, a kind and understanding person, and will always be constant source of inspiration to me. I have learned a great deal from his experience and intuition. I thank him for his valuable suggestions during my PhD research, as well as for granting me sufficient freedom to pursue my own ideas.

Secondly, I would like to express my gratitude to my good friend and research-mate Chunlei Liang. We started working on the project at almost the same time, and he was extremely supportive and helpful over the course of my research, particularly during the early stages. His motivation and support were instrumental in helping me complete my PhD in good time.

My gratitude also goes out to Professor Robert MacCormack, who had guided and supported me for more than a year after my Masters. It was a thoroughly enjoyable experience working with him. He has been a great mentor and more importantly a good friend. And I am sure that those who know him will not disagree when I say that he is one of the nicest people in academia today.

I am very grateful to Professor Sanjiva Lele for agreeing to be on my thesis reading committee, and for his valuable criticism and feedback. I would also like to thank Professor Peter Pinsky and Professor Gianluca Iaccarino for agreeing to be on my thesis defense committee, and for being so very cooperative.

I would be remiss if I did not acknowledge the financial support I received during the course of this project. This work was made possible by support from the Air Force Office of Scientific Research (AFOSR) and the National Science Foundation (NSF).

I would also like to thank my lab-mates and friends - Aaron, Divye, Edmond, Jender, Kui, Lala, Matt, Patrice, Peter, Rui and Yves. They definitely made the work experience more fun and enjoyable. They were also very helpful with issues related to my work. I wish them the best of luck in all their endeavors. I am also grateful for all the support from our kind and accommodating administrative staff - Carolyn, Jay, Lynn, Ralph, Robin - who have made student life at Aero/Astro smooth and easy.

On a more personal note, I sincerely enjoyed the many friendships I have made during my graduate life at Stanford . I would like to thank my good friends - Ankit, Jamit, Mamta, Marwah, Piyush, Roon, Sameer and Tanmay - for making my stay at Stanford so very enjoyable. I also had a great time with my friends in Raagapella, Crooked Mile and FMA, who shared my passion for music.

Last, but certainly not the least, I would like to thank my parents and sister for the support and encouragement they have given me throughout my education and PhD research. None of this would have been possible without their love and support.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

The field of Computational Fluid Dynamics has seen tremendous growth over the past few decades. This advance can be attributed to both the increasing power of computational resources as well as the progress made in the field of numerical solution techniques. As a result, numerical modeling has become an essential component of engineering design and analysis, especially in the field of fluid mechanics. While experimental measurements will always have a role in the design process, CFD offers significant advantages in terms of ease of use, cost and flexibility. Nowadays, commercial flow solvers are capable of solving large turbulent flow problems for complex geometries, up to engineering accuracy, with reasonable turn-around times. This has resulted in significantly reducing the time, resources and cost involved in the design of flow devices, by reducing the need for building and testing prototypes in the early stages of a design process. Nevertheless, despite the advances in the capabilities and usage of CFD, there is still ample room for progress.

In most industrial applications of computational fluid flow, simulations on unstructured meshes have been dominated by low order schemes. Almost all of the commercial software packages that allow unstructured mesh computations are restricted to spatially 2nd order accurate solutions. The popularity of low order methods may be attributed to several factors. Firstly, in many industrial applications,

the requirements on accuracy are not excessive and engineering accuracy is usually sufficient. Secondly, the work on low order schemes over the last half century have resulted in schemes that are robust and can handle complex flow phenomenon such as shocks and other discontinuities. Also, convergence acceleration techniques for such schemes are extremely efficient and well-understood.

However the classical second order accurate schemes fall short when computations require a high level of accuracy. This requirement is common in several modern flow applications such as Large Eddy Simulation (LES), Direct Numerical Simulation (DNS), Computational Aeroacoustics (CAA), or numerical turbulent combustion, to name a few. In LES, the use of low order methods can result in masking of subgrid-scale dissipation by numerical dissipation. In CAA, we require very low dissipation to track wave propagation over long distances. With turbulent combustion, the small scales of turbulence have a significant influence on mixing and combustion efficiency, and low order schemes are incapable of resolving these features accurately. The use of low order methods for such applications results in mesh requirements that are often prohibitive. In contrast, high order methods are less dissipative and possess superior wave propagation properties, and hence promise to provide higher accuracy with fewer degrees of freedom.

The most direct consequence of using high-order methods is the reduction of the impact of discretization error on the accuracy of the computed solutions. Low order schemes are generally 2nd order spatially accurate, i.e. discretization error is proportional to h^2, where h is the mesh-spacing. For high order methods, the discretization error is proportional to h^r, where $r > 2$, for sufficiently smooth problems. In the past decade, there has been a surge of research activities on high-order discretizations on unstructured meshes. The main motivation for these efforts is the expectation that high-order methods have the ability to achieve the required accuracy levels for flow problems with complex physics and geometry more efficiently. However these methods are still in the development stage and are not mature enough to be used for real industrial applications. They lack the robustness displayed by traditional low order schemes. Also, they lag behind in terms of computational machinery for efficient computations of complex real-world flow problems. Furthermore, there exist

a number of high-order methods for unstructured grids that are under development and we have not yet reached the stage where we can pick the optimal one.

High order accurate schemes have been used with structured grids quite successfully for a long time now. On structured grids, high order accuracy can be obtained relatively easily by extending the stencil used for solution reconstruction, as well as derivative computations. Extending the stencil is straightforward with structured grids as the information regarding the neighboring points or cells is readily available. An alternate approach is the use of Compact Difference Schemes which use a compact stencil, but approximate the leading terms in the truncation error to obtain high-order accuracy. The application of structured grid based high order methods is however, limited to relatively simple domains and geometries. When we have to deal with complicated geometries typical of industrial flow problems, the generation of structured meshes is very difficult and extremely time-consuming. On the other hand, unstructured meshes are more suited to applications with complex geometries, and the process of mesh generation is easier and often automated.

In contrast to structured meshes, it is not straightforward to obtain high-order accuracy by just extending the stencil for solution reconstruction using unstructured meshes. This is because we no longer have immediate accessibility of neighboring cell data. Moreover such reconstruction procedures on unstructured grids often lead to ill-conditioned or even singular linear algebraic systems, which cause serious degradation of numerical accuracy. Due to this extended stencil, efforts to parallelize such methods to run on multiple processors can pose certain difficulties. To avoid this shortcoming, focus shifted to a class of methods with high-order, locally discontinuous solution representations. Such methods approximate the solution as a polynomial of desired degree within each cell. However the solution may be discontinuous across the cell interfaces. Methods such as Discontinuous Galerkin (DG), Spectral Volume (SV) and Spectral Difference (SD) belong to this class of methods.

The DG method is arguably the most popular among the high-order methods for unstructured grids. This method has been under development since the 80's and has hence seen a good amount of progress both in terms of applicability as well as availability of relevant computational machinery. In a DG method, the Degrees of

Freedom (DOFs) are either the expansion coefficients for a given set of polynomial basis functions or solutions at selected locations within the element, and a Galerkin finite-element method is employed to update the unknowns within each cell. DG possesses favorable properties such as a firm mathematical basis and general non-linear stability for arbitrary cell shapes. However, its formulation is complicated, making it difficult to interpret physically. It is also computationally expensive due to the surface and volume integrals that have to be numerically evaluated. In the Spectral Volume (SV) method, the integral conservation law is used to update volume averages over subcells defined by a geometrically similar partition of each grid cell. As the order of accuracy increases, the partitioning for a 3D simplex cell requires the introduction of a large number of parameters, whose optimization to achieve convergence becomes increasingly more difficult. Also, the large number of interior facets, and the additional increase in the number of quadrature points for each facet increase the computational cost significantly.

The Spectral Difference Method is based on the differential form of the conservative equations. The absence of volume or surface integrals makes the SD method easier to implement in multiple dimensions than the DG and SV methods. For the same reason, the SD method is also more efficient than the latter. This was the main motivation behind choosing SD as the test-bed for further numerical experiments. In the SD method, the number of DOFs in each cell is the number of nodal values required to support a reconstruction of a given order of accuracy. Their locations are chosen so that a quadrature approximation for the volume integral exists at least to the same order of accuracy. The fluxes are calculated at a different set of nodes, whose number supports a reconstruction of one order higher, since the flux derivatives are used to update the conservative unknowns. They are located so that quadrature approximations for surface integrals over the cell boundaries exist to a required order of accuracy. In addition, the locations of the solution points and the flux points are chosen such that the integral conservation law is satisfied for the cell to the desired order of accuracy. If the points are distributed in a geometrically similar manner for all cells, the reconstruction and discretization become universal, and the residual can be expressed as the same weighted sums of the products of the local metrics and the

fluxes.

1.2 Challenges with using high order methods

One of the greatest challenges with using high-order methods is their slow convergence rate. High-order spatial discretization operators are usually much stiffer than their low-order counterparts. An issue that requires careful attention with high-order spatial methods is the design of efficient time marching techniques. Classical explicit time marching algorithms, such as explicit Runge-Kutta schemes, have upper limits for the time step based on stability considerations, which are prescribed by the Courant-Friedrichs-Lewy (CFL) condition. Such classical algorithms can be very inefficient in combination with high-order spatial schemes, for which the maximum time step tends to be very small. Such restrictively small time steps can be avoided by the use of implicit time marching algorithms, which are generally stable for much larger time-steps. However, with such algorithms, a nonlinear algebraic system must be solved at each time step. Designing an efficient solver for these systems is critical in order to extract the benefits of larger time-steps.

Another class of convergence acceleration techniques that has been used quite successfully is the multigrid method. Algebraic solvers usually eliminate the high frequency errors quite effectively, but the low frequency errors are damped very slowly. The classic multigrid algorithm involves interpolation from the fine mesh to coarser mesh levels. The low frequency errors on the fine mesh become high frequency on the coarse mesh, and are hence damped effectively. The p-multigrid or the multi-order method is just an extension of the classical h-multigrid to a high order setup. Here we use lower orders of polynomial representation instead of coarser mesh levels.

The solution of hyperbolic conservation laws often gives rise to flow discontinuities in the form of shocks or contact discontinuities. Numerical schemes designed to solve these partial differential equations (PDEs) must be capable of capturing any discontinuity that may arise in the solution. One of the greatest challenges with using high-order methods is their inability to handle flow discontinuities. When flows

involve steep gradients such as shock waves or contact surfaces, non-physical spurious oscillations arise that contaminate the solution in smooth regions of the flow, and often cause the simulations to go unstable. Higher order approximations are less dissipative than their low-order counterparts and hence it is typically necessary to add explicit dissipation in order to obtain a stable solution. But this has a negative effect on accuracy in the vicinity of the discontinuity. It may also degrade the resolution of turbulent scales due to excessive damping. The development of numerical algorithms that can capture discontinuities and also resolve the scales of turbulence in compressible turbulent flows remains a significant challenge.

There exists a wide spectrum of approaches to handle flow discontinuities, such as shock fitting, reconstruction methods, limiters and artificial viscosity. Shock fitting techniques involve determination of the shock location within the computational domain through analytical, experimental, or numerical means. The shock location is treated as a boundary condition of sorts within the computational analysis and higher-order accuracy can be attained away from the shock. However, shock fitting is often not viable, as we are solving for unknown flow fields and the position of the shock is not know *a* priori. The use of limiters is associated with some of the older and more successful classes of shock-capturing based on the Total Variation Diminishing (TVD) or the Local Extremum Diminishing (LED) criterion. In flow regions where limiting is active, the approximating polynomial is reduced to a piecewise-constant representation, leading to a solution that is total variation diminishing in the mean values of each element. However, previous efforts using limiters with SV and SD were found to suffer from convergence issues. Polynomial reconstruction methods such as ENO (Essentially non-oscillatory) and WENO (Weighted Essentially Non-Oscillatory) retain higher-order modes and utilize the additional degrees of freedom to yield sharp shock transitions. They adaptively choose a stencil (or weighted multiple stencils) to reconstruct a higher order polynomial from a set of local cell average values while eliminating spurious oscillations. However, the adaptation of such methods to unstructured mesh calculations is non-trivial.

The artificial viscosity approach involves the explicit addition of dissipation in the region of discontinuities by introducing viscous terms to the governing partial

differential equation. Viscosity that is on the order of the resolution length scale of the discretization smears out discontinuities until they can be well represented. To ensure consistency of the numerical approximation, this artificial viscosity must disappear as $h \rightarrow 0$ and not impact the solution in regions of smooth flow. Artificial viscosity/dissipation has been applied quite successfully for low order computations on unstructured grids as well as with high-order schemes on structured grids. Recent studies have applied artificial viscosity within a DG setup. The current work proposes the addition of smoothly varying artificial viscosity in the region of discontinuities.

The addition of artificial dissipation means that the solution is locally first order accurate in the vicinity of the discontinuity. The most obvious technique to improve the accuracy of computing flows with discontinuities would be to refine the mesh in the vicinity of the shock. Adaptive mesh refinement is a widely used and accepted strategy for improving the accuracy of a computational simulation while limiting the increase in computational cost. However, adaptively refining a quadrilateral mesh leads to the generation of hanging nodes. To deal with the non conforming solution polynomials on interfaces with hanging nodes, a mortar-element method is used. In a high order setup, it is also beneficial to adaptively vary the polynomial order p. Experiments with local order refinement are also conducted in the present work.

1.3 Motivation for the thesis

Despite the advance in CFD over the past few decades, the field of high-order methods for unstructured grids has not yet reached the level of maturity required to solve real flow problems on complicated geometries. The present thesis work makes an effort towards the realization of a flow solver that is high-order accurate but also efficient, robust and geometrically flexible at the same time. All the numerical computations have been conducted on 2D test cases, but the present numerical scheme, as well as the relevant computational tools that have been developed and implemented as part of this work, can be easily extended to 3D. The Spectral Difference method is relatively new and less mature as compared to the Discontinuous Galerkin Method. But in addition to being high order accurate, the SD method has a simple formulation

and efficient implementation, which essentially amounts to lesser computational effort as compared to other methods in its class. This is why we have chosen the SD method over other methods like DG or SV.

Due to their slow convergence rates, it becomes viable to use high order schemes only when there is some kind of convergence acceleration technique used. The p-multigrid and the implicit time stepping techniques have proved to be very effective on studies with DG, and we wish to investigate the improvement in convergence rates for the SD method. Another issue plaguing high-order methods is robustness - particularly their ability to deal with flow discontinuities. Earlier efforts with limiters on SD scheme reported issues with convergence to steady state. Also, shock capturing based on artificial viscosity has been used successfully with structured grid based high-order methods, and recently applied to DG. This motivated us to investigate an artificial viscosity based approach with the SD scheme, and design a formulation that is simple, efficient, robust and can be easily extended to 3D. Adaptive mesh and order refinement have great potential in reducing the computational effort required to reach a specified level of accuracy, particularly in the context of high-order formulations. Both mesh and order refinement in quadrilaterals result in non-conforming solution approximations at cell interface, which can be handled using a mortar element method. We look to enable the capability for adaptive mesh and order refinement within the solver, and also examine the resulting reduction in computational effort.

1.4 Outline of the thesis

The outline of the remainder of the thesis is as follows. Chapter 2 includes a survey of available literature on high order schemes and related topics such as convergence acceleration, artificial viscosity and adaptive mesh and order refinement. The partial differential equations that govern inviscid and viscous compressible flows that are solved in the present thesis, are described in chapter 3. Chapter 4 discusses the formulation, properties and implementation of the Spectral Difference (SD) method. The implementation of efficient time-marching schemes for the SD method, namely p-multigrid and implicit time stepping are addressed in Chapter 5. The next chapter

describes the salient features of the artificial viscosity approach used to compute flows with shocks. Chapter 7 looks at adaptive mesh and order refinement in the context of the SD scheme, and describes the mortar-element method used to enable this. Finally, conclusions are drawn and future perspectives are considered in Chapter 8.

Chapter 2

Literature survey

2.1 High order methods

The earliest high-order accurate numerical methods were the Spectral Methods [37, 20], where the solution of a differential equation is approximated over the entire domain using a high-order expansion. Choosing the expansion functions properly, one can achieve arbitrarily high-order accuracy. However, because of the global nature of the expansion functions, spectral methods are typically limited to very simple domains with simple boundary conditions. Motivated by the prospect of obtaining high accuracy with the greater geometric flexibility, researchers in the early 1980s introduced the p-type finite element method. In the p-type finite element method, the grid spacing, h, is fixed, and the interpolation order, p, is increased to drive the error down. In 1981, Babuska et al. [5] applied this method to elasticity problems. In 1984, Patera [92] introduced a variant of the p-type method, known as the Spectral Element method, and used it to solve the incompressible Navier-Stokes equations for flow in an expanding duct. Korczak and Patera [68] later extended this method to more general, curved geometries.

There has been a good amount of progress with high order schemes on structured meshes. In this context, Lele [69] introduced up to tenth-order accurate Compact Difference schemes in 1992. This work was extended and applied by Visbal and Gaitonde [123], wherein they used the high-order compact difference method to solve

the compressible Navier-Stokes equation on curvilinear meshes. Zingg et al. [135] showed that high-order compact difference schemes are more efficient, in terms of number of nodes required for drag computation to a desired accuracy level, than typical second-order finite differences. Another notable contribution was the Essentially Non-Oscillatory (ENO) [42, 41] schemes introduced in 1987. WENO schemes were developed later to improve upon ENO schemes, in Liu et al. [73] and Jiang and Shu [58]. Advantages of WENO schemes over ENO included the smoothness of numerical fluxes, better steady-state convergence, and better accuracy using the same stencils.

In an unstructured, finite volume context, Barth [8] introduced the k-exact reconstruction method, which is based on a least-squares reconstruction procedure that requires an extended stencil. Furthermore, ideas from the ENO methods were also extended to unstructured grids by many researchers [1, 29, 89]. WENO schemes were extended to unstructured grids by Friedrich [35], and Hu and Shu [48].

Finite element methods are attractive for achieving high-order accuracy because, for smooth problems, the order of accuracy is controlled by the order of the solution and test function spaces. However, it is well known that standard, continuous Galerkin methods are unstable for the convection operator [122]. This brings us to locally discontinuous finite element based methods, the most notable of which was the Discontinuous Galerkin (DG) method.

The DG method is probably the most mature and developed high order accurate method for unstructured grids. It was introduced in 1973 by Reed and Hill [100] for a steady conservation law, namely the neutron transport equation. It was first used for unsteady advection laws by Van Leer [119] in 1978. A breakthrough in the application of DG methods to nonlinear hyperbolic problems was made by Cockburn and Shu [22, 23, 24], who introduced the Runge Kutta Discontinuous Galerkin (RKDG) method. A comprehensive overview of these RKDG methods can be found in a review article by Cockburn and Shu [25]. Independent of the above work, Allmaras [2], and Allmaras and Giles [3] developed a second-order DG scheme for the 2-D Euler equations, their method being an extension of van Leer's method of moments for the 1-D linear wave equation.

More recently, many researchers [9, 10, 11, 19, 24, 25, 88] have applied DG methods to diffusion problems. One procedure, pioneered by Bassi and Rebay [9, 10] and generalized by Cockburn and Shu [24, 25], is to rewrite a second-order equation as a first-order system and then discretize the first-order system using the DG formulation. These mixed formulations became popular as BR1 and BR2, the latter producing a nearest neighbor stencil. The generalization of the first-order system idea by Cockburn and Shu leads to the so-called Local Discontinuous Galerkin methods (LDG). In multiple dimensions, LDG has an extended stencil, but a recent modification of LDG by Peraire and Persson [94] known as Compact Discontinuous Galerkin (CDG) recovers a more compact stencil while retaining the attractive properties of LDG. A quadrature-free DG formulation, which is more efficient than the standard formulation in terms of computational cost, was developed by Atkins and Shu [4].

Another locally discontinuous approach, based on the finite volume scheme is the Spectral Volume Method. The basic methodology of the SV method was first presented by Wang [126] in 2002, along with its application to one-dimensional scalar hyperbolic conservation laws. Wang and Liu [128] extended the SV method to two-dimensional scalar equations and also studied different limiting strategies to capture discontinuities. The same authors presented the extension of the SV method to one-dimensional systems of conservation laws, along with an optimization study of the SV partitions, in [129]. Subsequently, the SV method was applied to two-dimensional inviscid flow problems, by Wang et al. [132]. A very important issue with all high-order methods, namely the appropriate treatment of curved wall boundaries, was addressed for the 2D SV method by Wang and Liu [130], by using a high-order geometric mapping of the SV cells near such boundaries. Liu et al. [75] applied the SV method to 3D computational electromagnetics (CEM) problems. The formulation of the SV method for the N-S equations was originally developed and presented by Sun et al. [110]. A further contribution to the SV method for diffusive problems was made by Kannan et al. [62], who investigated different formulations for the discretization of diffusive terms with the SV method. Comparisons of the SV method with the DG method were made by Sun and Wang [109] and Zhang and Shu [134]. Analogous to the quadrature-free formulation of the DG method, a quadrature-free

formulation of the SV method was developed by Harris et al. [40].

The origins of the method now known as the Spectral Difference method date back to 1996. Kopriva and Kolias [67] and Kopriva [65] introduced their formulation for the solution of the 2D compressible Euler equations on unstructured quadrilateral meshes, which they called the 'Conservative Staggered-Grid Chebyshev Multi-Domain method'. In 1998, Kopriva [66] extended the method to viscous formulations. A general formulation of this approach on simplex cells, was given in 2006 by Liu et al.[74], who first called it the Spectral Difference method, and applied it to two-dimensional scalar conservation laws and Computational ElectroMagnetic (CEM) equations. Wang et al. [131] extended the SD method for simplex cells was to the 2D Euler equations. It was further extended to the 2D N-S equations by May and Jameson [84], and Wang et al. [60]. Sun et al. [111] implemented the SD method on hexahedral cells for the 3D N-S equations. Different approaches for the discretization of the diffusive terms in the N-S equations with the SD method, based on similar approaches that were developed for the DG method, were investigated by Van den Abeele et al. [117]. Van den Abeele et al. [115] also proved an interesting property of the SD method, namely that it is independent of the positions of its solution points. They also performed an extensive study of the stability and accuracy properties of the SD method. The equivalence of SD and SV in 1D was reported in Van den Abeele et al.[116]. Huynh [49] proposed a set of 1D SV and SD schemes based on Legendre-Gauss quadrature points and proved that these are stable for arbitrary orders of accuracy. Recently, Jameson [55] obtained a theoretical proof that the SD method is stable for all orders of accuracy in a Sobolev norm provided that the interior flux points are located at the zeros of the corresponding Legendre polynomial.

A new, unifying, high-order method for unstructured grids, recently proposed by Huynh [49, 50], who named it flux reconstruction approach, and by Wang and Gao [127], who referred to it as lifting collocating penalty (LCP) method, combines properties of the DG method and the SV and SD methods. It has a simple, easily interpretable, point-wise formulation, which does not involve any integrals and consequently, the evaluation of the residuals is relatively cheap. With a certain choice of parameters, the method can be made linearly equivalent to the DG, SV or SD

method. It then also has the same linear stability properties as these methods.

2.2 Convergence acceleration techniques for high order methods

High-order accurate methods are in general much stiffer than low-order methods. High-order explicit time marching algorithms, such as the Runge-Kutta (R-K) schemes, when combined with high-order spatial schemes, suffer from restrictively small time steps due to the Courant-Friedrichs-Lewy (CFL) stability limit. This is especially true for viscous problems, where cells with high aspect ratios are needed to resolve boundary layers. These restrictive CFL limits can be overcome by using implicit time marching schemes. Several implicit time-stepping algorithms have been used in literature. Element Jacobi methods were used for convergence acceleration of the DG method by Helenbrook and Atkins [44]. Newton-GMRES solvers with preconditioners were used in combination with DG schemes by Bassi and Rebay [13], and in combination with SV and SD schemes by Van den Abeele et al. [117, 118]. Line-implicit solvers developed by Mavriplis [82], were applied to DG by Fidkowski et al. [32]. A matrix-free Krylov method was applied to DG schemes by Rasetarinera and Hussaini [99] and to SD schemes by May et al. [83]. Nonlinear Lower Upper Symmetric Gauss-Seidel (LU-SGS) solvers (see Jameson and Yoon [57],Chen and Wang [21]), were used with SD schemes by Sun et al. [112, 113], Van den Abeele et al. [117] and Premasuthan et al. [98], and with SV schemes by Kannan et al. [62], Parsani et al. [91] and Haga et al. [38].

The use of geometric multigrid for compressible flow equations was pioneered by Ni [87](1981) and Jameson [52] (1983), and has been widely used in CFD since then. The p-multigrid algorithm, in which the coarser solution approximations are obtained by switching to lower-degree polynomials, was introduced by Ronquist and Patera [103] in 1987. Further developments were reported by Maday and Munoz [79]. Since then, Bassi and Rebay [12], Helenbrook et al. [44, 45] and Fidkowski et al. [32] have applied it to the DG method. It was also used with the SV method by Van den

Abeele et al. [114], Parsani et al. [91] and Kannan et al. [62]. Application of the p-multigrid to the SD method were reported by May et al. [83], Liang et al. [70] and Premasuthan et al. [98],

2.3 Shock Capturing with artificial viscosity

The classical approach of using artificial viscosity/dissipation for shock capturing was pioneered by von Neumann and Richtmeyer [124]. One of the earlier efforts involving the use of artificial viscosity with compressible flow equations was by Baldwin and MacCormack [6]. The concept of flexible addition of artificial viscosity/dissipation has been used very successfully by Jameson et al. [51, 56, 53, 54], thus producing non-oscillatory and sharp resolution of shocks for structured and unstructured finite volume calculations. Jameson used a blended diffusion operator in schemes like JST (Jameson-Schmidt-Turkel) and SLIP(Symmetric Limited Positive), where the artificial dissipation is third order in smooth regions of the flow and first order when there is a discontinuity.

Cook and Cabot proposed an artificial viscosity method for high-order centered differencing schemes, wherein a spectral-like high-wavenumber biased artificial viscosity and diffusivity were dynamically added [27, 26]. This was followed up with work by Fiorina and Lele [34], on high-order compact difference schemes, wherein artificial diffusivity was added in addition to artificial viscosity. Kawai and Lele [63] extended the method to non-uniform and curvilinear meshes. This method involves the dynamic addition of grid-dependent localized transport coefficients such as artificial bulk viscosity, shear viscosity and artificial conductivity where needed. The application of this form of artificial viscosity has been limited to structured grid computations.

Other forms of artificial viscosity have been applied to high-order unstructured grid calculations. Persson and Peraire [95] introduced a p-dependent artificial viscosity and demonstrated that higher-order representations and a piecewise-constant artificial viscosity can be combined to produce sub-cell shock resolution. Barter and Darmofal [7] proposed shock-capturing using a combination of higher-order PDE-based artificial viscosity and enthalpy-preserving dissipation operator. Both the above

methods were proposed for high-order Discontinuous Galerkin (DG) discretizations. The present author investigated the use of an artificial viscosity based approach with the Spectral Difference Method [96, 97]. The formulation and implementation of this approach are discussed in detail in Chapter 6.

2.4 Adaptive mesh and order refinement

Compressible flow calculations on unstructured meshes with adaptive mesh refinement have become increasingly popular over the last few decades. Early efforts to enhance solution procedures using adaptive grid refinement include those by Berger and Jameson [15], Mavripilis and Jameson [81], Holmes et al. [47, 46], Lohner [76] and Peraire et al. [93]. With the increased use of high-order methods, order or p-refinement has been used in conjunction with h-refinement. Previously, refinement criterion was based on error indicators depending on the solution gradients, curvatures, residuals or other relevant flow features. The recent past has seen the development of output based adaptive methods that rely on the solution of an adjoint problem based on the output of interest [121, 14, 36]. These have been extended to DG formulations by Hartmann and Houston [43], Fidkowski and Darmofal [31], and Wang and Mavripilis [125]. Quite recently, Fidkowski and Roe [33] proposed an entropy based approach for mesh refinement, with the comparative advantage that no adjoint equation need be solved. Adaptive mesh refinement was used with the Spectral Volume approach by Harris and Wang [39]. Kopriva [65] demonstrated the implementation of adaptive mesh and order refinement for the multi-domain spectral method on quadrilateral meshes. The present author uses a mortar element method similar to the one described by Kopriva to enable mesh and order refinement, the details of which are described in Chapter 7. This quadrilateral mesh refinement has been combined with artificial viscosity to produce sharp resolution of shocks using the Spectral Difference method [97].

Chapter 3

Governing Equations

This section presents the partial differential equations that are solved as part of this work. The flow of a viscous compressible fluid is governed by a set of equations called the Navier Stokes Equations. The 2D Navier Stokes equations in conservative form can be written as,

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0 \tag{3.1}$$

where Q is the vector of conserved variables. F and G are the total flux vectors in the x and y direction respectively. F and G can be split into inviscid and viscous parts, $F = F_i + F_v$ and $G = G_i + G_v$. If the viscous flux components are not included, equation 3.1 corresponds to the Euler equations, which are used to model inviscid fluid flows.

The conservative variables and the inviscid components of the fluxes are given by,

$$Q = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix} F_i = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u\left(\rho E + p\right) \end{pmatrix} G_i = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v\left(\rho E + p\right) \end{pmatrix}$$

where ρ is the density of the fluid, u and v are the cartesian velocity components of the flow, p is the pressure, and E is the specific total energy. The specific total energy E is related to the specific internal energy, by the relation,

$$E = e + \frac{u^2 + v^2}{2} \tag{3.2}$$

The specific internal energy e is given by $e = C_v T$ where T is the temperature of the fluid, and C_v is the specific heat at constant volume. For an ideal gas, we have the ideal gas relation between the density, pressure and temperature, $p = \rho RT$, where R is the Specific Gas Constant which is about $287 Jkg^{-1}K^{-1}$ for air. The above mentioned equations and relations constitute a closed system of equations, that can be solved. Another useful relation is the one between pressure and the specific total energy.

$$p = (\gamma - 1)\rho \left(E - \frac{u^2 + v^2}{2} \right) \tag{3.3}$$

The viscous flux vectors can be written as

$$\begin{aligned} F_v &= - \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{yx} \\ u\tau_{xx} + v\tau_{yx} + \kappa \frac{\partial T}{\partial x} \end{pmatrix} \\ G_v &= - \begin{pmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ u\tau_{xy} + v\tau_{yy} + \kappa \frac{\partial T}{\partial y} \end{pmatrix} \end{aligned} \tag{3.4}$$

where the τ's are components of the shear stress tensor, and κ is the thermal conductivity of the fluid. The shear stress tensor is related to the velocity gradients as given below.

$$\begin{aligned} \tau_{xx} &= 2\mu \frac{\partial u}{\partial x} + \beta \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) \\ \tau_{yy} &= 2\mu \frac{\partial v}{\partial y} + \beta \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) \\ \tau_{yx} &= \mu \left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right) \\ \tau_{xy} &= \mu \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) \end{aligned} \tag{3.5}$$

where μ is the dynamic (shear) viscosity coefficient, and β is the bulk viscosity coefficient. The latter is related to the viscous stress caused by a volume change. However,

under the Stokes' hypothesis, the bulk viscosity is related to the dynamic viscosity as $\beta = -(2/3)\mu$, and the trace of the shear stress tensor vanishes.

Dimensional analysis shows that the compressible viscous flow over a particular geometry is characterized by three dimensionless parameters. The first dimensionless parameter is the free stream Mach number, which is the ratio of the free stream velocity to the speed of sound at free stream conditions.

$$M_\infty = \frac{U_\infty}{c_\infty} \tag{3.6}$$

The second dimensionless parameter is the Reynolds number (Re). The Reynolds number of a flow is given by

$$Re = \frac{\rho_\infty U_\infty L_c}{\mu} \tag{3.7}$$

where L_c is the characteristic length for the flow in concern, and μ is the dynamic viscosity coefficient. The Reynolds number can be interpreted as the ratio of the inertial forces and viscous forces, and is indicative of viscous effects on flow. If Re is low, the viscous stresses dominate and as a result the flow will be laminar. If the Re is high, the inertial stresses dominate, and the flow may be turbulent.

The third dimensionless parameter is the Prandtl number, which is given as

$$Pr = \frac{\mu c}{\kappa} \tag{3.8}$$

It is a measure of the ratio between the momentum diffusivity and the thermal diffusivity. It is more a property of the fluid than a measure of the flow state. The Prandtl number for air is about 0.72.

Chapter 4

Spectral Difference Method

The present formulation for the Spectral Difference method is based on quadrilateral meshes in 2D. It is known that simplex cells offer greater flexibility and ease of mesh generation. However quadrilateral meshes were chosen over triangular meshes due to two main reasons. Firstly, at the time of solver development, there did not exist a stable configuration for SD method of orders higher than three on triangular meshes. However, on quad meshes the SD method was stable upto arbitrary orders of accuracy. Also, quad meshes provide greater accuracy with lesser number of cells in the boundary layer regions. The formulation of the equations for the 2D spectral difference scheme on unstructured quadrilateral meshes is similar to the formulation of Kopriva [66] on quadrilateral meshes, and Sun et al [111] on hexahedral grids.

4.1 Formulation on quadrilateral meshes

Consider the unsteady compressible 2D Navier Stokes equations in conservative form, as described in equation 3.1. It is assumed that the computational domain is divided into non-overlapping quadrilateral elements, that are body-conforming at boundaries. In order to handle curved boundaries, high order iso-parametric elements are used. Thus linear elements are used in the interior of the domain, and quadratic or cubic elements are used adjacent to curved boundaries.

To achieve an efficient implementation, all elements in the physical domain (x, y)

are transformed into a standard square element, $0 < \xi < 1$, $0 < \eta < 1$. The transformation can be written as:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \sum_{i=1}^{K} M_i(\xi, \eta) \begin{pmatrix} x_i \\ y_i \end{pmatrix} \tag{4.1}$$

where K is the number of vertices used to define the physical element, (x_i, y_i) are the cartesian coordinates of those vertices, and $M_i(\xi, \eta)$ are the shape functions. The metrics and the Jacobian of the transformation can be computed for the standard element. The governing equations in the physical domain are then transformed into the computational domain, and the transformed equations take the following form:

$$\frac{\partial \tilde{Q}}{\partial t} + \frac{\partial \tilde{F}}{\partial \xi} + \frac{\partial \tilde{G}}{\partial \eta} = 0 \tag{4.2}$$

where $\tilde{Q} = |J| \cdot Q$ and

$$\begin{pmatrix} \tilde{F} \\ \tilde{G} \end{pmatrix} = |J| \begin{pmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{pmatrix} \begin{pmatrix} F \\ G \end{pmatrix} \tag{4.3}$$

In the standard element, two staggered sets of points are defined, namely the solution points and the flux points, illustrated in figure 4.1. The solution points are the locations where the nodal values of the conservative variables Q are specified. Flux points are the locations where the nodal values of fluxes F are computed. The number of flux points is chosen such that they can support a polynomial reconstruction of degree one higher than the solution polynomial. This is necessary to ensure that the flux derivatives (spatial) are polynomials of the same degree as the solution, since the flux derivatives are used to update the solution. There are two sets of flux points - family 1 corresponding to the flux points in the i-direction, and family 2 corresponding to the flux points in the j-direction. The solution unknowns or degrees of freedom in the SD method are the transformed conservative variables at the solution points.

In order to construct a degree $(N-1)$ polynomial in each coordinate direction, solution values at N points are required. The solution points in 1D are chosen to be the Chebyshev-Gauss points defined by:

$$X_s = \frac{1}{2}\left[1 - cos\left(\frac{2s-1}{2N} \cdot \pi\right)\right], s = 1, 2, \cdots, N. \tag{4.4}$$

The flux points were selected to be Legendre-Gauss quadrature points plus the two end points 0 and 1, as suggested by Huynh [49, 50], and theoretically confirmed by Jameson [55]. Choosing $P_{-1}(\xi) = 0$ and $P_0(\xi) = 1$, we can determine the higher-degree Legendre polynomials as

$$P_n(\xi) = \frac{2n-1}{n}(2\xi - 1)P_{n-1}(\xi) - \frac{n-1}{n}P_{n-2}(\xi) \tag{4.5}$$

The locations of these Legendre-Gauss quadrature points are the roots of equation $P_n(\xi) = 0$. They were found to be more stable than the Gauss-Lobatto flux points proposed earlier, and produce more accurate solutions for high-order spectral difference schemes.

Using the solutions at N solution points in 1D, a degree $(N-1)$ polynomial can be built using the following Lagrange basis defined as:

$$h_i(X) = \prod_{s=1, s\neq i}^{N} \left(\frac{X - X_s}{X_i - X_s} \right) \tag{4.6}$$

Similiarly, using the fluxes at $(N+1)$ flux points, a degree N polynomial can be built for the flux using a similar Lagrange basis defined as:

$$l_{i+1/2}(X) = \prod_{s=0, s\neq i}^{N} \left(\frac{X - X_{s+1/2}}{X_{i+1/2} - X_{s+1/2}} \right) \tag{4.7}$$

The reconstructed solution for the conserved variables in the standard element is just the tensor products of the two one-dimensional polynomials,

$$Q(\xi, \eta) = \sum_{j=1}^{N} \sum_{i=1}^{N} \frac{\tilde{Q}_{i,j}}{|J_{i,j}|} h_i(\xi) \cdot h_j(\eta) \tag{4.8}$$

Similarly, the reconstructed flux polynomials take the following form:

$$\begin{aligned} \tilde{F}(\xi, \eta) &= \sum_{j=1}^{N} \sum_{i=0}^{N} \tilde{F}_{i+1/2,j} \cdot l_{i+1/2}(\xi) \cdot h_j(\eta), \\ \tilde{G}(\xi, \eta) &= \sum_{j=0}^{N} \sum_{i=1}^{N} \tilde{G}_{i,j+1/2} \cdot h_i(\xi) \cdot l_{j+1/2}(\eta) \end{aligned} \tag{4.9}$$

Notice that $\tilde{F}$ is reconstructed using the flux values at the family 1 flux points, and $\tilde{G}$ is reconstructed using the values at the family 2 flux points.

The solution polynomial is only element-wise continuous, but discontinuous across cell interfaces. Therefore, for the flux points at the cell interface, there exist left and right solution states and the flux is not uniquely defined. As a result, we replace the inviscid fluxes by numerical fluxes, and an approximate Riemann solver is employed to compute the common numerical flux at the interface flux points. These numerical fluxes are responsible for coupling the solutions in two adjoining cells and providing the necessary numerical dissipation to stabilize the numerical method. Using one-dimensional Riemann solvers to compute these numerical fluxes also ensures conservation, because the normal component of the flux vector on each face is identical for the two cells sharing that face. In the present study, we have used the Rusanov solver [104] (or Scalar Diffusion [53]) as well as Roe flux [102] as the approximate Reimann solver to compute the interface inviscid fluxes.

The algorithm to compute the inviscid flux derivatives can be summarized as follows:

1. Given the conservative variables at the solution points, the conservative variables are computed at the flux points using equation 4.8.

2. The inviscid fluxes are computed at the interior flux points.

3. The inviscid fluxes at the element interfaces are computed using the approximate Riemann solver.

4. The derivative of the fluxes are computed at the solution points by differentiating the flux polynomial in each coordinate direction.

$$\begin{aligned} \left(\frac{\partial \tilde{F}}{\partial \xi}\right)_{i,j} &= \sum_{r=0}^{N} \tilde{F}_{r+1/2,j} \cdot l'_{r+1/2}(\xi_i), \\ \left(\frac{\partial \tilde{G}}{\partial \eta}\right)_{i,j} &= \sum_{r=0}^{N} \tilde{G}_{i,r+1/2} \cdot l'_{r+1/2}(\eta_j) \end{aligned} \tag{4.10}$$

The viscous flux is a function of both the conserved variables and their gradients. Therefore, the solution gradients have to be calculated at the flux points. In the present work, the average approach described in reference [66] is used to compute

the viscous fluxes. The procedure to compute the viscous fluxes can be described as follows.

1. At the element interfaces, find the average of left and right values of Q_f; $\overline{Q_f} = \frac{1}{2}(Q_f^L + Q_f^R)$. For interior flux points, $\overline{Q_f} = Q_f$. Appropriate boundary conditions are applied at flux points on boundary edges.

2. Evaluate ∇Q at the solution points from $\overline{Q_f}$, where $\nabla Q = \left\{ \begin{array}{c} Q_x \\ Q_y \end{array} \right\}$ and $Q_x = \frac{\partial Q}{\partial \xi}\xi_x + \frac{\partial Q}{\partial \eta}\eta_x$, etc. $\frac{\partial Q}{\partial \xi}$ and $\frac{\partial Q}{\partial \eta}$ can be computed using an equation similar to 4.10.

3. Reconstruct ∇Q to the flux points, apply appropriate boundary conditions for boundary flux points, and average them on the element interfaces as $\overline{\nabla Q_f} = \frac{1}{2}(\nabla Q_f^L + \nabla Q_f^R)$

4. Use $\overline{Q_f}$ and $\overline{\nabla Q_f}$ in order to compute viscous flux vectors at the flux points.

4.2 Testing and Validation of the 2D SD solver

In this section, the results obtained from the testing and validation of the 2D SD flow solver are presented. Firstly, inviscid cases test cases are considered. The supersonic vortex test case and the inviscid flow past circle and airfoil are discussed in this context. This is followed by the validation of the viscous flow cases. The test cases demonstrated include the planar Couette flow test case, Taylor-Couette flow case, and viscous flow past a circle.

It should be mentioned that all explicit time-marching calculations for steady flows have been done using a Jameson type four-stage Runge Kutta scheme (RK4) [56], which is 2nd order accurate in time. For unsteady problems, we have used a 4th order accurate, strong-stability-preserving five-stage Runge-Kutta scheme [108] to advance in time.

Supersonic vortex flow

This test case is used to verify the implementation of the inviscid component of the 2D solver and assess the order of accuracy of the Spectral Difference method used. The supersonic vortex flow problem is one of the few non-trivial problems governed by the steady compressible 2D Euler equations for which a smooth analytical solution is known. The inviscid, isentropic, supersonic flow of a compressible fluid between concentric circular arcs presents a flow where the velocity varies inversely with radius. The expression for density as a function of radius r is given by:

$$\rho\left(r\right)=\rho_i\left\{1+\frac{\gamma-1}{2}M_i^2\left[1-\left(\frac{r_i}{r}\right)^2\right]\right\}^{\frac{1}{\gamma-1}} \tag{4.11}$$

where M_i and r_i are the Mach number and the radius at the inner arc. In the present calculation, the Mach number, density and pressure at the inner radius r_i are specified to be 2.25, 1 and $1/\gamma$ respectively. The inner and outer radii are 1 and 1.384 units respectively. A curved inviscid wall BC is used as wall boundary condition. The zero-gradient extrapolation boundary is employed for the exit. The numerical solution to this problem is computed for the 2nd, 3rd and 4th order SD method on successively refined grids. All the computations are initialized using constant density and pressure. The L^2-norm of density error is evaluated.

The four meshes used in the computation were of sizes 10×4, 15×6, 30×12, and 60×24. A sample 15×6 mesh is shown in figure 4.2(a). Figure 4.2(b) shows the density contours in the flow field obtained by the 3rd order SD method. The details of the order calculation and verification are shown in Table 4.1. The table clearly indicates that the SD method applied to the steady compressible Euler equations exhibits a full order of convergence on smooth solutions. Figure 4.3(a) provides the details of the spatial accuracy of the SD method for different orders for this numerical experiment. Figure 4.3(b) shows the L^2-error of the SD method at different orders plotted against the number of degrees of freedom (DOFs). One can clearly see that a higher order SD method requires a lesser number of degrees of freedom than a lower order SD method to achieve the same accuracy.

Inviscid Subsonic flow past a circle

In this section, the results obtained for 2D steady, subsonic flow around a circle at Mach number M_∞=0.2, are presented. The computations have been performed on a 32×32 grid. This test case demonstrates the significance of higher order representation of boundary elements to obtain high order accurate solutions.

Figure 4.4(a) show the Mach number contours using 4th order SD with linear representation of boundary. This solution is very inaccurate, as is evident from the non-physical "boundary layer" which develops along the solid wall, and by the associated "wake" in the downstream region of the circle. Moreover the computations do not converge to a steady solution due to the unsteadiness of the non-physical wake. Figure 4.4(b) shows the Mach number contours using the 4th order SD method with cubic boundary representation. It clearly indicates that the use of higher order representation for the boundary results in a smoother and more accurate solution.

Inviscid flow past airfoil

The inviscid subsonic flow at $M_\infty = 0.5$ past a NACA0012 airfoil at zero degree angle of attack is studied. The airfoil surface is represented using a high-order boundary representation - cubic splines in this case. The outer boundary is about 25 chords away from the airfoil mid-chord. Figure 4.5(b) shows the pressure contours for a 3rd order SD computation on a computational mesh with 1280 cells (shown in figure 4.5(a))

This test case was chosen to study drag convergence for the 3rd and 4th order SD computations. The inviscid drag on airfoil at zero degrees angle of attack is zero. Hence, the computed drag coefficient value was taken as the error in the computations. Four levels of successively refined meshes were used for the SD calculations. These meshes were obtained from FLO-82. FLO-82 is a 2D finite-volume inviscid flow solver developed by Dr. Antony Jameson, which is spatially 2^{nd} order accurate. This also facilitated the comparison of results obtained using the SD solver with those obtained from FLO-82 for identical flow conditions using similar meshes. FLO-82 is considered industry standard for 2D inviscid flow computations [120].

Table 4.2 shows the computed drag for the 3rd and 4th order Spectral Difference

calculation. As expected, the computed order for drag convergence with mesh refinement for the 3rd order computation is around 3. Similarly, the 4th order computation exhibits drag convergence of order close to 4. Table 4.3 shows the results obtained using FLO-82, which indicate that the drag error drops at an order close to 3 for the most part. A look at the drag convergence vs. degrees of freedom (shown in figure 4.6 (a)) indicates that the 3rd order SD method does not perform better than FLO-82 in terms of accuracy for given number of DOF's. However, the 4th order SD achieves higher accuracy for the same number of DOF's as compared to FLO-82.

However, it must be noted this added accuracy comes at a price, since the computational time required to converge to steady state is generally higher for the higher order computation. Figure 4.6(b) shows a comparison of the computational times required for 4th order SD versus FLO-82. It must be mentioned that for both codes, the L2-norm of density residual was allowed to drop to 10^{-10}. Also, the 4th order computations used Implicit Symmetric Gauss Siedel scheme (described in the next section) for time-stepping. It is clear that for moderate levels of accuracy, the higher order method takes longer to converge to steady state than FLO-82. The trends however indicate that beyond a certain level of accuracy, it is more beneficial to use the 4th order SD. This particular case of steady, inviscid flow past a streamlined body (airfoil) might not be strong candidate for using high-order accurate schemes. The benefits of high-order accuracy in terms of overall computational effort are more obvious for unsteady flows with vortical interactions, turbulent fluctuations etc.

Planar Couette flow

The planar Couette flow test case was used to test and validate the viscous component of the flow solver, and to confirm the order of accuracy of viscous flow computations. This problem models the viscous flow between two parallel plates located at $y = 0$ and $y = H$. The plate at $y = 0$ is stationary and kept at a temperature T_0, and the plate at $y = H$ is moving at speed U in the x-direction and is at fixed temperature T_1. Under the assumption of constant viscosity coefficient, this problem has an exact solution, which can be expressed as

$$\begin{aligned} u(y) &= \frac{U}{H}y, v = 0, \\ T(y) &= T_0 + \frac{y}{H}(T_1 - T_0) + \frac{\mu U^2}{2k} \cdot \frac{y}{H}\left(1 - \frac{y}{H}\right), \\ p &= const, \rho = \frac{p}{R \cdot T} \end{aligned} \tag{4.12}$$

The numerical solution to this problem was computed for the 2nd, 3rd and 4th order SD methods on successively refined grids. All the computations are initialized using constant density and pressure. Isothermal wall BC was used for top and bottom walls, and periodic BC was used for left and right boundary. The L^2-error of the density is evaluated. The four meshes used in the computation were of sizes 2×1, 4×2, 8×4, and 16×8. Figure 4.7 shows the density contours in the flow field obtained by 3rd order SD method using the 4×2 grid. The details of the order calculation and verification are shown in Table 4.4. The table clearly indicates that the SD method applied to the steady compressible Navier Stokes equations exhibits a full order of convergence on smooth solutions.

Taylor-Couette flow

In this example, the numerical order of accuracy is validated against the analytical solution for the compressible Taylor Couette flow. This test problem was taken from a recent paper presented by Michalak and Ollivier-Gooch [86]. The Reynolds number is 10 based on the inner cylinder tangential velocity and its radius (=1 unit). The temperature and pressure are prescribed on the inner cylinder giving a Mach number 0.5. The outer cylinder is fixed and an adiabatic wall boundary condition is employed. A grid with 24x2 cells is shown in figure 4.8(a). Two other finer grids are obtained using successive grid refinements in both directions.

A high order curved wall boundary is used for inner and outer cylinders. Quadratic boundary representation is used for the 3^{rd} order computation and cubic representation is used 4^{th} order computation. A steady solution of Mach number contour obtained by the SD method is shown in figure 4.8(b). We obtain desired numerical order for the L2-norm of error in angular velocity as shown in Table 4.5. The viscous

solver demonstrates the required order of accuracy for the 3^{rd} and 4^{th} order computations. This case also validates the higher order representation of the curved boundary.

Viscous flow past a circle

The present case involves viscous flow past a circle at Re=100 and free-stream Mach number of 0.2. Figure 4.9 shows the computational grid used. There are 32 cells around the circumference of the circle. The computation for this case is performed using the fifth-order SD method and a cubic curved wall boundary condition is employed for the circular surface. Dirichlet boundary condition is used for the inlet and fixed-pressure is used for the outlet boundary condition. Symmetry boundary conditions are applied on the two lateral sides. The initial condition was set to free-stream condition. A 4^{th} order computation was also made and the difference between the fluctuating lift coefficient $C_l^{'}$ and the drag coefficient $C_d^{'}$ predicted by the 4th-order (total DOFs 21,376) and the fifth-order SD methods (total DOFs 33,400) were found to be less than 2%. In the following, we only present the results obtained by the fifth-order SD method.

Table 4.6 reports the comparison between the present computation of compressible viscous flow at Mach number 0.2 and other numerical and experimental studies for incompressible viscous flow at the same Reynolds number 100. The Strouhal number predicted by the SD method on a mesh with degree-of-freedom 33,400 is identical to the one predicted by Sharman et al.[105] and the experimental value measured by Williamson [133]. The computed force coefficients also seem to compare reasonably well with previous results. It must be mentioned that the previous studies involved incompressible flow calculations. The results show that the SD method accurately predicts the unsteady physics for low Reynolds number flow past a cylinder.

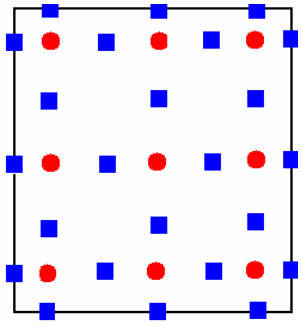
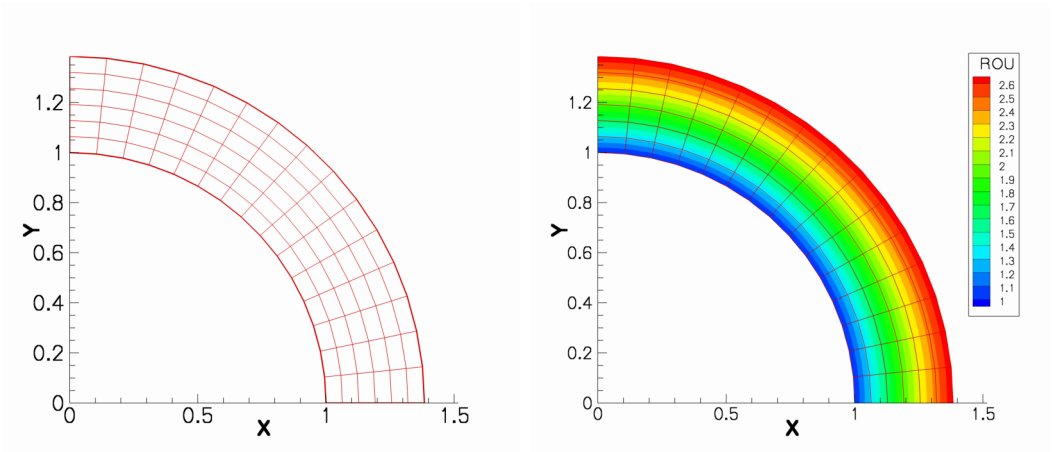
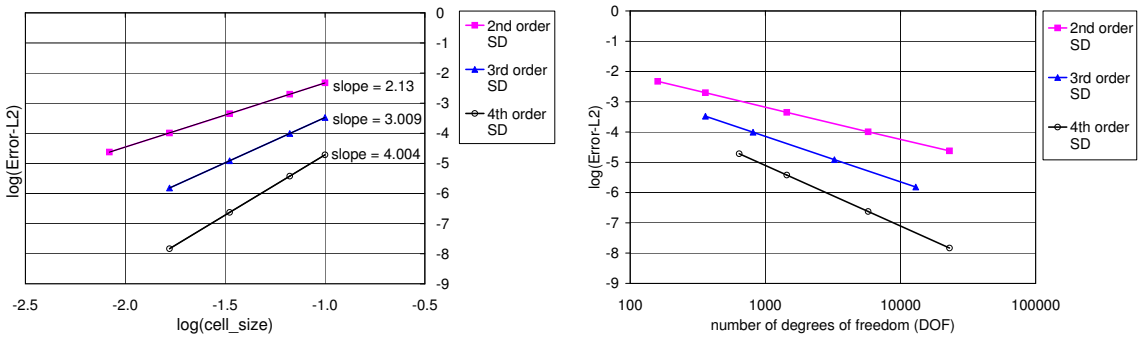


Figure 4.1: Position of solution (circles) and flux (squares) points on standard square element for 3rd order SD



(a) (b)

Figure 4.2: (a) 15×6 mesh for supersonic vortex flow test case; (b) Density contours using 3rd order SD.

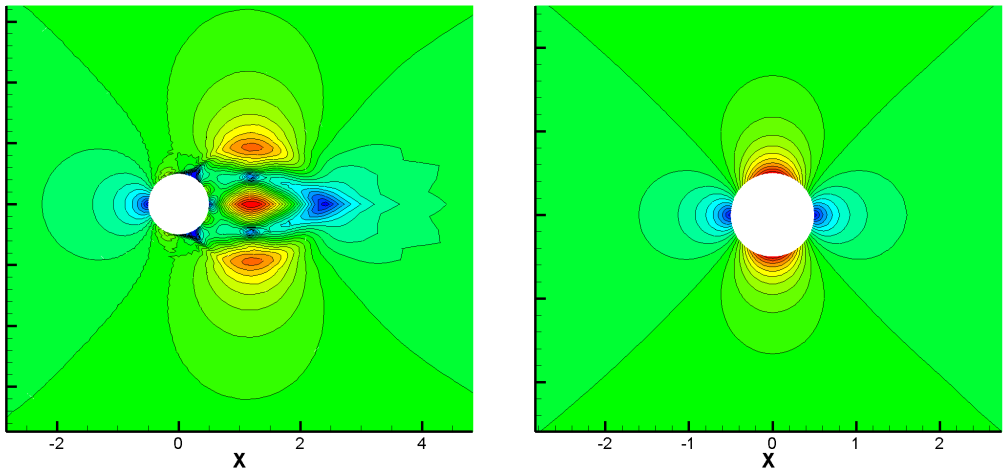


(a) (b)

Figure 4.3: Accuracy summary for the supersonic vortex flow case for SD method using 2nd, 3rd and 4th order solution approximations (a) L^2 error vs cell-size; (b) L^2 error vs degrees of freedom.

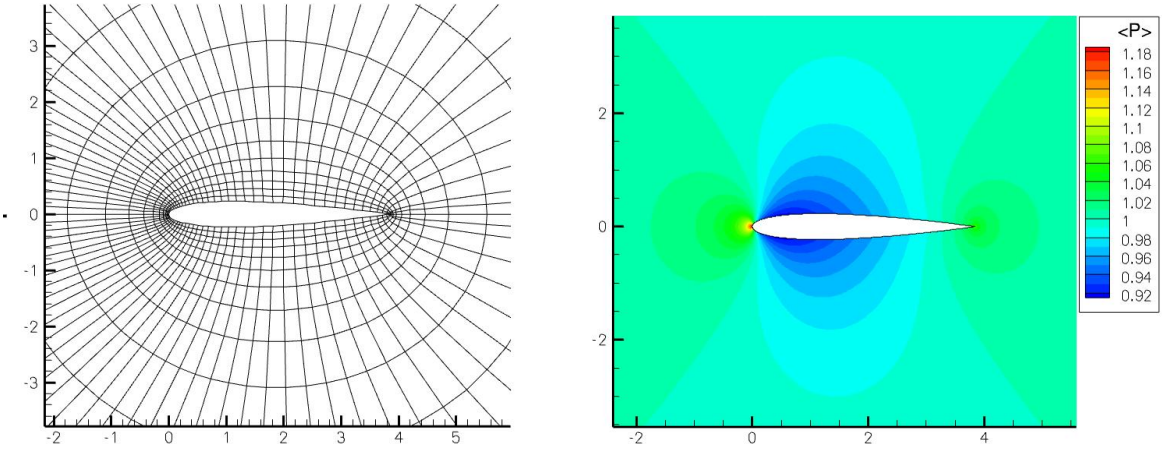
No. of elements	No. of DOFs	L2-error	Order
2nd order SD			
40	160	4.7249E-03	-
90	360	1.9881E-03	2.135
360	1440	4.4721E-04	2.152
1440	5760	1.0196E-04	2.133
3rd order SD			
40	360	3.3393E-04	-
90	810	9.8833E-05	3.003
360	3240	1.2242E-05	3.013
1440	12960	1.5230E-06	3.007
4th order SD			
40	640	1.9238E-05	-
90	1440	3.7883E-06	4.008
360	5760	2.3651E-07	4.002
1440	23040	1.4743E-08	4.004

Table 4.1: L^2 errors and orders of accuracy for inviscid supersonic vortex flow



(a) (b)

Figure 4.4: Mach contours for inviscid flow past a circle (a) 4th order SD with linear boundary (b) 4th order SD with cubic boundary



(a) (b)

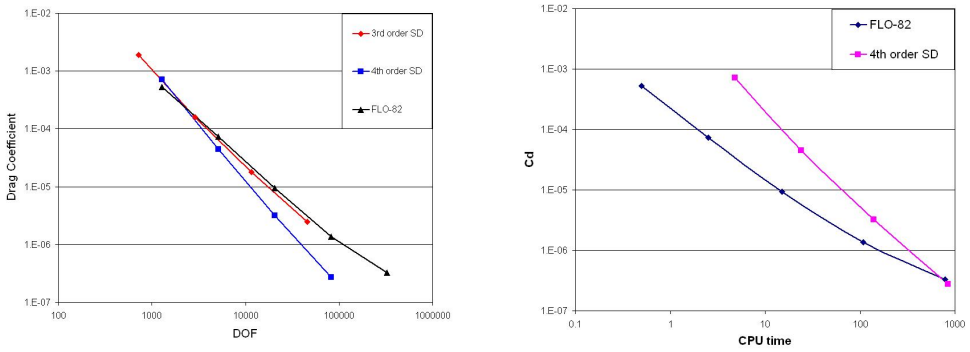
Figure 4.5: Inviscid flow around NACA0012 (a) Computational grid (80×16); (b) non-dimensional pressure contours for 3rd order SD.

No. of elements	No. of DOFs	Cd	Order
3rd order SD			
80	720	1.901E-03	-
320	2880	1.583E-04	3.59
1280	11520	1.780E-05	3.14
5120	46080	2.492E-06	2.85
4th order SD			
80	1280	7.189E-04	-
320	5120	4.482E-05	4.00
1280	20480	3.212E-06	3.79
5120	81920	2.746E-07	3.55

Table 4.2: Drag coefficient and orders of accuracy for inviscid subsonic flow past airfoil using SD

Mesh size	No. of DOFs	Cd	Order
FLO-82			
20x4	1280	5.293E-04	-
40x8	5120	7.259E-05	2.87
80x16	20480	9.396E-06	2.95
160x32	81920	1.357E-06	2.79
320x64	327680	3.274E-07	2.05

Table 4.3: Drag coefficient and orders of accuracy for inviscid subsonic flow past airfoil using FLO-82



(a) (b)

Figure 4.6: Comparison of high-order SD with FLO-82 for inviscid flow past airfoil case (a) Drag coefficient vs. degrees of freedom (b) Drag coefficient vs. computational time

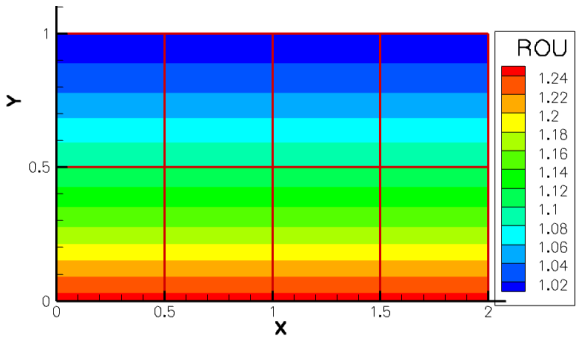
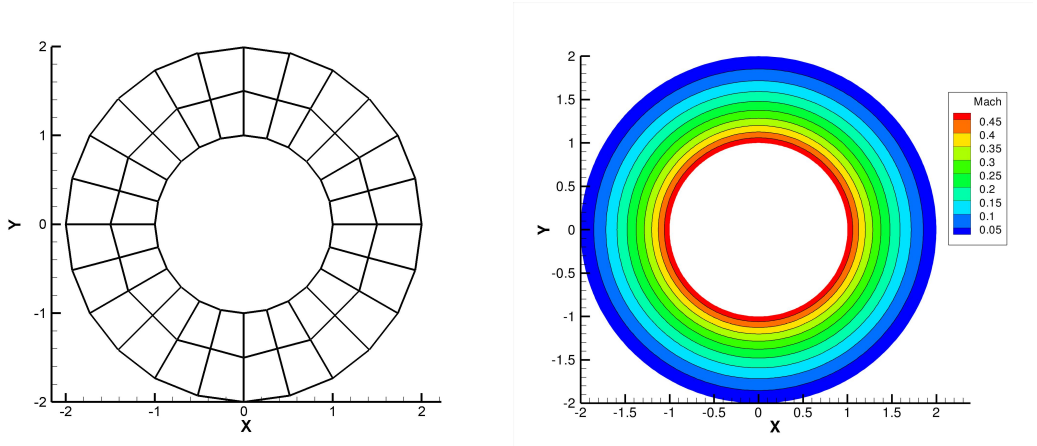


Figure 4.7: Density contours for Couette flow problem using 3rd order SD (4×2 grid)

No. of elements	No. of DOFs	L2-error	Order
2nd order SD			
2	8	1.4180E-02	-
8	32	3.3520E-03	2.081
32	128	9.1210E-04	1.878
128	512	2.4350E-04	1.905
3rd order SD			
2	18	1.4783E-03	-
8	72	1.5199E-04	3.282
32	288	1.6525E-05	3.201
128	1152	1.7991E-06	3.199
4th order SD			
2	32	1.9784E-04	-
8	128	1.1827E-05	4.064
32	512	7.9780E-07	3.890
128	2048	4.7330E-08	4.075

Table 4.4: L^2 errors and orders of accuracy of viscous Planar Couette flow

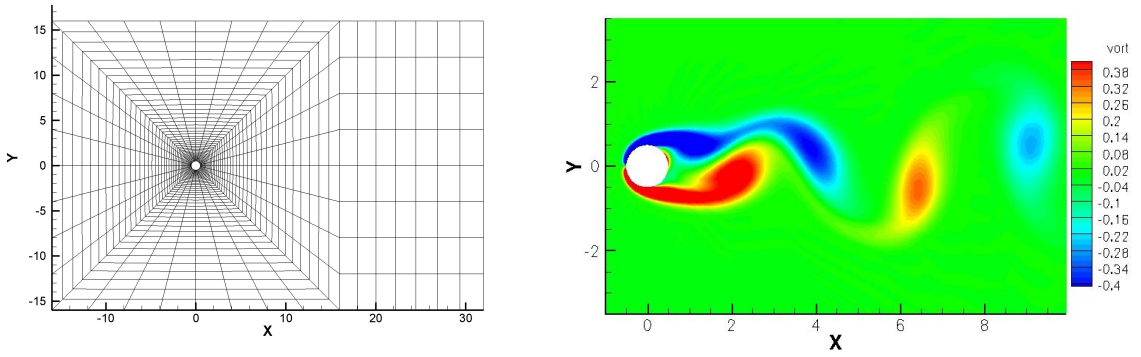


(a) (b)

Figure 4.8: Taylor-Couette flow (a) Computational grid (b) Mach contours for 3rd order SD.

No. of elements	No. of DOFs	L2-error	Order
3rd order SD			
48	432	8.896E-04	-
192	1728	1.002E-04	3.15
768	6912	1.084E-05	3.21
4th order SD			
48	768	1.4815E-04	-
192	3072	1.0036E-05	3.88
768	12288	6.5746E-07	3.93

Table 4.5: L^2 errors and orders of accuracy for Taylor-Couette flow



(a) (b)

Figure 4.9: Viscous flow around cylinder (a) Computational grid (b) Vorticity contours for 4th order SD.

Investigator	Present	Sharman [105]	Mene [85]	Kang [61]	Ding [28]
Re no.	100	100	100	100	100
nodes	1,336	14,441	13,696	62,127	23,033
blockage	0.0312	0.02	0.047	-	-
$C_l^{'}$	0.232	0.23	-	0.32	0.287
$\overline{C_d}$	1.365	1.33	1.37	1.33	1.356
$C_d^{'}$	0.0086	0.0064	-	-	0.01
St. no.	0.164	0.164	0.165	0.165	0.166

Table 4.6: Comparison of the present results against results from previous work on flow over a cylinder at Reynolds number 100

Chapter 5

Convergence Acceleration

High-order spatial discretization operators are usually much stiffer than their low-order counterparts, thus leading to slow convergence rates. The design of efficient time marching techniques is an issue that requires careful attention with high-order spatial methods. Classical explicit time marching algorithms, such as explicit Runge-Kutta schemes, have upper limits for the time step based on stability considerations. Such classical algorithms can be very inefficient in combination with high-order spatial schemes for which the maximum time step tends to be very small. In the present chapter we investigate the use of the p-Multigrid method as well as implicit time-stepping to accelerate convergence.

All explicit time-marching calculations for steady flows have been done using a Jameson type four-stage Runge Kutta scheme (RK4) [56], which is 2nd order accurate in time. For unsteady problems, we have used a 4th order accurate, strong-stability-preserving five-stage Runge-Kutta scheme [108] to advance in time.

5.1 p-Multigrid method

The p-multigrid method is a natural extension of geometric multigrid methods to high-order accurate formulations. Classical geometric multigrid method utilizes the fact that on the fine mesh, the high frequency errors are eliminated quickly but the low frequency errors take longer to get damped. The solution is then transferred to

coarser meshes so that the low frequency errors (corresponding to fine mesh) become high frequency errors on the coarse mesh and are eliminated effectively. The corrected solution is then transferred back to the fine mesh. In contrast to geometric multigrid that operates on a sequence of successively coarser meshes, the p-multigrid method operates on a sequence of solution approximations of different polynomial orders. Even though this method has been referred to as the p-multigrid method, it might be more appropriate to call it the multi-p or multi-order method.

The key idea of the p-multigrid method is to solve the nonlinear equations using a lower order polynomial such that "smooth becomes rough" and low frequencies act like high frequencies. Classical p-multigrid method begins with a two-level process. First, iterative relaxation is applied using the higher order polynomial, thus basically reducing high-frequency errors. Then, a "coarse-grid" correction is applied, in which the smooth error is determined at the lower polynomial level. This correction is interpolated to the higher polynomial level and used to correct the existing higher order solutions. Applying this method recursively to solve at lower polynomial levels leads to a multi-level p-multigrid algorithm. It should be noted that the p-multigrid method offers the flexibility of switching between higher and lower polynomial levels without changing the actual geometrical nodal grid points. Therefore, the need to generate multiple meshes is eliminated.

The following steps summarize the standard two-level scheme with a V-cycle at p and $p-1$ levels:

Firstly, we are trying to solve $R_p(Q_p) = r_p$, where $R_p(Q_p)$ corresponds to the residual of current solution Q_p. r_p is the right hand side (RHS) and equals 0 at the highest polynomial level p.

1. Apply smoothing iterations at level p to improve the solution Q_p.

2. Compute the defect at level p.

$$d_p = r_p - R_p\left(Q_p\right) \tag{5.1}$$

3. Restrict the latest solution and the defect to the lower approximation level $p-1$

$$Q_{p-1}^0 = I_p^{p-1}\left(Q_p\right) \tag{5.2}$$

4. Compute the rhs of equation at level $p-1$.

$$r_{p-1} = R_{p-1}\left(Q^0_{p-1}\right) + I^{p-1}_p d_p \tag{5.3}$$

5. Apply smoothing iterations at level $p-1$ to improve Q_{p-1}

6. Compute the correction at level $p-1$.

$$C_{p-1} = \bar{Q}_{p-1} - Q^0_{p-1} \tag{5.4}$$

7. Extrapolate the correction from level $p-1$ to update the solution at level p.

$$\tilde{Q}_p = Q_p + I^p_{p-1} C_{p-1} \tag{5.5}$$

8. Improve Q_p by application of iterative smoother

In this study a multi-level V-cycle has been used to drive the iterations. To accomplish the communication between different levels, we use restriction and prolongation operators derived from cardinal basis functions of the solution points for the starting p-level. Currently the Jameson type Runge-Kutta (RK4) explicit scheme is employed as a smoother at all levels. The p-multigrid has been applied to only problems with a steady state solution.

The p-multigrid has been used by other researchers to accelerate convergence for the DG method [32, 44, 77]. For SD schemes, the p-multigrid has been used for inviscid flows on triangular grids [70, 84]. The present work studies the performance and effectiveness of p-multigrid with quadrilateral grids using up to four-level multigrid cycles. Earlier studies for Euler flow calculations have indicated that the use of explicit smoothing iterations at all levels adversely affects the performance of the p-multigrid. This has been attributed to the severely anisotropic (hyperbolic) nature of the Euler equations and the isotropic (elliptic) nature of p-multigrid iterations [77]. Efforts will be made to address this issue with regards to the viscous flow equations. The speed-up obtained using p-multigrid with explicit RK smoothers, is compared to the speed-up obtained using the implicit LU-SGS scheme.

5.2 Implicit Backward Euler Scheme with LU-SGS

When using explicit time marching schemes, there exist restrictions on the maximum allowable time-step, which are defined by the CFL condition. Furthermore, the use of high-order spatial discretizations result in making these restrictions more stringent. The use of implicit time-marching methods is advantageous as they allow much larger time-steps. However, implicit time marching schemes generally involve a large non-linear algebraic system, which needs to be solved every time iteration n. An efficient algorithm for such systems is thus essential.

The present section discusses the non-linear Lower-Upper Symmetric Gauss Seidel (LU-SGS) algorithm which was introduced for second order finite volume schemes by Jameson and Yoon [57]. The implicit LU-SGS scheme has been used with the Spectral Difference Method for inviscid flows on triangular meshes [70], and also been used for viscous computations on hexahedral meshes [111]. A similar approach is used here for implementation with quadrilateral meshes.

At each cell c, the governing equations can be discretized using the backward Euler difference formula to give

$$\frac{Q_c^{n+1} - Q_c^n}{\Delta t} - [R_c(Q^{n+1}) - R_c(Q^n)] = R_c(Q^n) \tag{5.6}$$

Let $\Delta Q_c = Q_c^{n+1} - Q_c^n$ be the change in the solution over the implicit time step. Linearizing the residual, we obtain

$$R_c(Q^{n+1}) - R_c(Q^n) \approx \frac{\partial R_c}{\partial Q_c}\Delta Q_c + \sum_{nb \neq c} \frac{\partial R_c}{\partial Q_{nb}}\Delta Q_{nb}, \tag{5.7}$$

where nb indicates all the neighboring cells contributing to the residual of c. Therefore, the fully linearized equations can be written as

$$\left(\frac{I}{\Delta t} - \frac{\partial R_c}{\partial Q_c}\right)\Delta Q_c - \sum_{nb \neq c} \frac{\partial R_c}{\partial Q_{nb}}\Delta Q_{nb} = R_c(Q^n). \tag{5.8}$$

However, it costs too much memory to store the LHS implicit Jacobian matrices. We employ a LU-SGS scheme to solve equation 5.8, thus using the most recent solution in the neighboring cells,

$$\left(\frac{I}{\Delta t} - \frac{\partial R_c}{\partial Q_c}\right)\Delta Q_c^{(k+1)} = R_c(Q^n) + \sum_{nb \neq c} \frac{\partial R_c}{\partial Q_{nb}} \Delta Q_{nb}^{*}. \tag{5.9}$$

Note that the superscript "$*$" refers to the most recent solution when doing the forward and backward sweeps of the LU-SGS scheme, and the superscript "k+1" denotes the new solution that is obtained after a particular sweep.

The left-hand side matrix given by,

$$D = \left(\frac{I}{\Delta t} - \frac{\partial R_c}{\partial Q_c}\right) \tag{5.10}$$

is the element cell matrix. For a 3rd order computation in 2D, the size of this matrix is $(4 \times 3 \times 3)^2 = 576$ elements. The computation of the element matrix involves the computation of the cell Jacobian matrix $\frac{\partial R_c}{\partial Q_c}$ which is not a trivial operation. It must be pointed out that for the Navier-Stokes equation the residual in a cell depends not only on its immediate neighbors but also the neighbors' neighbors and hence the computation of the summation term on the right-hand side of equation 5.9 can be quite computationally intensive. This can be avoided however, by further linearization to eliminate the off-diagonal Jacobian matrices $\frac{\partial R_c}{\partial Q_{nb}}$, thus giving a diagonally dominant system.

$$\left(\frac{I}{\Delta t} - \frac{\partial R_c}{\partial Q_c}\right)\Delta^2 Q_c^{(k+1)} = R_c(Q^*) - \frac{\Delta Q_c^*}{\Delta t}. \tag{5.11}$$

where $\Delta^2 Q_c^{(k+1)} = \Delta Q_c^{(k+1)} - \Delta Q_c^* = Q_c^{(k+1)} - Q_c^*$.

Equation 5.11 is solved with a direct LU decomposition solver and by sweeping symmetrically forward and backward through the computational grid. For steady flow calculations, the term $\frac{\Delta Q_c^*}{\Delta t}$ in the right-hand side can be neglected, leading to faster convergence.

The cell Jacobian $\frac{\partial R_c}{\partial Q_c}$ is computed using a numerical approach as shown below

$$\frac{\partial R_c}{\partial Q_c} = \frac{R_c(Q_{nb}, Q_c + \epsilon) - R_c(Q_{nb}, Q_c)}{\epsilon} \tag{5.12}$$

A numerical approach for Jacobian computation has been previously used [111, 70]. Even though this approach is easy to implement, the computational effort involved is large since each variable at each solution point has to be changed, and the

residuals recomputed. Numerical tests have shown that it is not necessary to compute the Jacobian matrix at every iteration. In practice it needs be computed every 40-100 iterations. This technique has been called matrix-freezing and it does not degrade convergence to steady state.

5.3 Implicit second order Backward Difference formula

The implicit formulation described in the previous section is first order accurate in time. Hence it would not be sufficient for time-accurate unsteady flow simulations. To deal with unsteady flow problems, we use the second order Backward Difference Formula, which is second order accurate in time.

To discretize the set of equations in time, we start with the second order Backward Difference Formula with a variable time step.

$$\frac{1+2\tau_n}{1+\tau_n}Q_c^{n+1} - (1+\tau_n)\,Q_c^n + \frac{\tau_n^2}{1+\tau_n}Q_c^{n-1} = \Delta t_n R_c^{n+1} \tag{5.13}$$

where $\tau_n = \frac{\Delta t_n}{\Delta t_{n-1}}$, and n is the index of the time iteration.

Following a process of discretization and linearization similar to that described in the previous section, we obtain a final equation of the form

$$\left(\frac{I}{\Delta t_n} - \frac{1+\tau_n}{1+2\tau_n}\frac{\partial R_c}{\partial Q_c}\right)\Delta^2 Q_c^{(k+1)} = \frac{\tau_n^2}{\Delta t_n\,(1+\tau_n)}\left(Q_c^n - Q_c^{n-1}\right) + \frac{1+\tau_n}{1+2\tau_n}R_c(Q^*) - \frac{\Delta Q_c^*}{\Delta t}. \tag{5.14}$$

where $\Delta^2 Q_c^{(k+1)} = \Delta Q_c^{(k+1)} - \Delta Q_c^* = Q_c^{(k+1)} - Q_c^*$.

The method of Symmetric Gauss Seidel sweeps has been used to solve this system of equations. The number of symmetric sweeps per time-step required to maintain time-accuracy has been investigated and is presented in the next section.

5.4 Application of p-multigrid and Implicit time-stepping

Subsonic inviscid flow past bump

This test case involves subsonic flow in a channel with a 10% thick circular bump on the bottom. The length of the channel is 3 units and its height 1 unit. The inlet Mach number is 0.5. This test case has been previously used for p-multigrid computations with DG formulations [77] as well as with SD formulations [70]. The mesh which contains 3201 grid-points, 3072 elements in total and 32 nodes to resolve the bump, is depicted in figure 5.1(a). It must be mentioned that for this grid, a 3rd order computation would involve $(3072 \times 9 =)$ 27648 computational nodes. The circular surface of the bump is represented as a quadratic boundary. The pressure contours obtained using the 4th order SD scheme with 4-level p-multigrid scheme is shown in figure 5.1(b), and is almost identical to those obtained in references [77, 70]. All p-multigrid calculations for the bump case are done using RK4 scheme for smoothing at all levels. For the implicit LU-SGS calculation, 11 forward and backward sweeping iterations are done at each time-step, and the jacobian matrix is computed every 40 time-steps.

A comparison of the convergence histories using the explicit RK4 scheme shows that the convergence deteriorates drastically when the SD order is increased from 1 to 2. For 3rd and 4th order SD, the RK scheme failed to converge. These convergence trends for higher order SD using RK time-stepping for the subsonic bump case have been documented earlier [77, 112]. The application of the p-multigrid method results in superior convergence characteristics for second, third and fourth order SD schemes. However, the Implicit LU-SGS time-stepping is able to obtain fastest convergence.

Figure 5.2(a) shows the convergence acceleration in terms of the CPU time for the second order SD scheme. A 2-level V-cycle p-multigrid method has been used with single smoothing iteration at the higher level (p1) and 2 smoothing iterations at the lower level (p0). (It should be noted that p0 corresponds to 1st order SD, p1 to 2nd order and so on.) It is evident that speed-up is obtained using the 2-level p-multigrid cycle. The speed-up using implicit time-stepping is about 5 times compared to the

2-level p-multigrid cycle.

In the absence of p-multigrid both 3rd and 4th order fail to converge. However, the use of p-multigrid results in favourable convergence characteristics as seen in figure 5.2(b). For the 3rd order SD, a 3 level V-cycle is used with 1-1-2-1-0 smoothing iterations at p2-p1-p0-p1-p2 levels. For the 4th order SD case, a 4-level V-cycle with 1-1-1-2-1-1-0 smoothing iterations at p3-p2-p1-p0-p1-p2-p3 levels is used. A comparison of p-multigrid and implicit speed-up for 3rd and 4th order is presented in figure 5.3(a) and (b).

The effect of changing the V-cycle on the convergence of 3rd order SD was studied. As expected the p2-p1-p0 3-level multigrid was found to give best results, as seen in figure 5.3(a). However, the p2-p0 2-level (3,1 (a)) V-cycle produces comparable convergence acceleration. This can be attributed to the fact that for a single level, p0 has the fastest convergence. The p2-p1 2-level cycle has poor convergence characteristics due to the slow convergence at p1 level. The speed-up using implicit scheme is about 7 times faster than the fastest p-multigrid cycle for the residual to drop to 1e-8.

In the case of the 4th order SD method, the 4-level V-cycle (4,3,2,1 p-MG) with 1-1-1-2-1-1-0 iterations at p3-p2-p1-p0-p1-p2-p3 is found to be most effective for convergence acceleration, as seen in figure 5.3(b). The 3-level V-cycle with 1-1-2-1-0 iterations at p3-p1-p0-p1-p3 (4,2,1 p-MG) gives the next best convergence. The use of p3-p2-p0-p2-p3 (4,3,1 p-MG) and the p3-p2-p1-p2-p3 (not shown in figure) 3-level cycles does not show good convergence characteristics. This could be attributed to the fact that two of the levels p3 and p2 both have very poor convergence. The implicit scheme is about 10 times faster than the fastest p-multigrid cycle.

Inviscid flow past airfoil

The details of this test-case have been described earlier in the section 4.2. The present computation uses the 80×16 mesh with 1280 cells. Figure 5.4(a) shows the convergence speed-up using the p-multigrid and implicit schemes for 3rd order SD computation. The 3-level cycle with 1-1-2-1-0 smoothing iterations at p2-p1-p0-p1-p2 levels is found to give best convergence. It is about 1.5 times faster than the 2-level p-multigrid cycle with 1-2-0 smoothing iterations at p2-p1-p2 levels. For a

residual drop to 1e-8, a p-multigrid speed up of about 7-8 is obtained as compared to the explicit RK4. In comparison, an implicit speed up obtained is about 300 times compared to explicit RK4.

It must be mentioned that the explicit RK4 takes about 240,000 iterations to converge to a residual of 10^{-8} at a CFL number of 0.5. The 3-level p-multigrid with 1-1-2-1-0 smoothing iterations at the p2-p1-p0-p1-p2 levels converges in about 14,500 V-cycles. The implicit scheme takes about 290 iterations to converge to the same level of residual. Each iteration involved 5 forward-backward sweeps and the gradient matrix was computed every 40 iterations. The CFL limit at the beginning of the computation was set to 0.5 and was linearly ramped over the course of the simulation. There was no upper limit on the CFL.

Figure 5.4(b) shows the convergence speed-up using the p-multigrid and implicit schemes for 4^{th} order SD computation. The 4-level p-multigrid cycle with 1-1-1-2-1-1-0 iterations at p3-p2-p1-p0-p1-p2-p3 is found to give a convergence speed up of about 9-10 times compared to the explicit RK4 case. The 2-level and 3-level p-multigrid cycles were found to give poor speed-up compared to the 4-level cycle in this case. The convergence using the implicit scheme is about 3 orders of magnitude faster than that of the explicit RK case. For the 4th order computation, the CFL limit for the explicit calculation is about 0.2, and for the implicit calculation is about 3000.

Planar Couette flow

The p-multigrid method and implicit scheme were used for convergence acceleration for the planar Couette flow test case. All multigrid calculations were done using the four stage RK scheme for time marching at all levels. It must be mentioned that for these computations, the multilevel p-multigrid cycle used has a single smoothing iteration at the higher levels and 2 smoothing iterations at the lowest level (p0). Also, the computations were done on the 8×4 grid corresponding to 128, 288 and 512 computational nodes for the 2nd, 3rd and 4th order SD respectively.

In figure 5.5(a), we see that for a second order computation, the application of p-multigrid accelerates convergence by about 3 times. The implicit speed-up obtained is about 13 times compared to explicit RK4. For the third order computation, the

2-level V-cycle (3,1 p-MG) is found to be more effective for convergence acceleration than the 3-level V-cycle (3,2,1 p-MG) (see figure 5.5(b)). In this case, the application of p-multigrid is able to reduce the computation time by a factor of about 5, and the implicit scheme reduces the computational time by about 40. In the case of the fourth order computation, the 3-level V-cycle (4,2,1 p-MG) gives the best convergence characteristics (see figure 5.5(c)). A speed-up of about 8 is obtained for 4th order SD. The implicit speed-up is about 110 compared to the explicit RK.

Viscous flow past NACA0012 airfoil

The results obtained for viscous fluid flow past NACA0012 airfoil at zero degree angle of attack are reported here. The free stream Mach number is 0.5 and the flow Reynolds number is 5000. These flow conditions are identical to one of the test-cases in Bassi and Rebay [9]. The 120×24 C-grid (see figure 5.6) with 78 points on the airfoil surface is used in these calculations. Figure 5.7(a) shows the Mach contours obtained for a 3rd order SD calculation. The surface C_p distribution compares well with that obtained by Bassi and Rebay for a 4th order DG computation (see figure 5.7(b)). The coefficient of drag $C_D = 0.531$ obtained with the 3rd order SD compares well with $C_D = 0.535$ obtained with the 3rd order DG calculation. Also, the skin friction coefficient indicates the separation point at 0.84c, which compares well with the value predicted by Bassi and Rebay.

Figure 5.8(a) shows the convergence speed-up using the p-multigrid and implicit schemes for 2nd order SD computation. The 2-level p-multigrid gives a speed-up of about 3.5 for the viscous flow case. The implicit scheme once again gives a much higher speed up of about 40 times compared to the explicit RK. For the 3rd order case, a 3-level p-multigrid V-cycle is used with 1-1-2-1-0 smoothing iterations at the p2-p1-p0-p1-p2 levels. A speed-up of about 8 is obtained until the residual drops to about 1e-7. As the residual drops further the p-multigrid performance deteriorates slightly as shown in figure 5.8(b). In comparison, the implicit scheme still obtains a very high speed up of more than 2 orders of magnitude.

It must be mentioned that for the viscous 3rd order calculation, explicit RK4 takes about 10^6 iterations to converge to a residual of 10^{-8} at a CFL number of 0.5. The

3-level p-multigrid with 1-1-2-1-0 smoothing iterations at the p2-p1-p0-p1-p2 levels converges in about 27,000 V-cycles. The implicit scheme takes about 380 iterations to converge to the same level of residual. Each iteration involved 5 forward-backward sweeps and the gradient matrix was computed every 40 iterations. The CFL upper limit for the implicit calculation was of the order of 10^5.

In comparison to the inviscid flow past airfoil case, the performance of the p-multigrid for the viscous flow past airfoil shows relatively unsatisfactory convergence characteristics. As indicated in earlier studies using p-multigrid for viscous Discontinuous Galerkin [32] and Spectral Volume [62], this issue may be resolved by using implicit smoothers at the different p-multigrid levels.

Also, when using the implicit LU-SGS time-stepping, the speed-up obtained for the inviscid airfoil flow case is much higher than that for the viscous case. This could possibly be due to two reasons. Firstly, the implicit LU-SGS implementation requires much higher computational effort for viscous flow calculations. For inviscid flow, the residual within an element (cell) depends only on its immediate neighbors, whereas for the viscous case, residual computations involve communication with its neighbors' neighbors. This leads to increased computational effort and time for computation of the gradient matrix as well as residual based on most updated solution of neighbors. Another explanation could be that the LU-SGS performance degrades when using stretched grids.

Unsteady Euler Vortex Advection case

The propagating euler vortex is an inviscid model problem with known analytical solution (refer to Erlebacher et al. [30] for details of the analytical expressions). The vortex passively propagates through the domain at a speed and direction defined by the specified Mach number and propagation angle. The strength of the vortex decays exponentially for large r, and the profile depends on parameters such as the vortex strength and the radius of the vortex core.

The domain size is rectangular with size 20×15 units and has periodic BC's in the vertical and horizontal directions. The mach number is chosen to be $M_\infty = 0.5$, the angle of vortex propagation with respect to the x-axis is $tan^{-1}0.5$. The parameters

for vortex propagation include the vortex strength which was chosen to be 0.5, and the radius of the vortex core which was chosen to be 1.0. The vortex is initially positioned at the origin. A fifth order SD simulation was used with implicit BDF2 time-stepping. The mesh (with 3072 cells) used is shown in figure 5.9(a) along with the initial density contours. The purpose of the present case was to verify the temporal accuracy of Implicit BDF2 time-stepping for computing unsteady flows. Hence, it was necessary to use a fine mesh with high solution polynomial order, so that the temporal discretization errors dominate over the spatial discretization errors.

The exact analytical solutions were available as a function of space and time. We used them to compute the temporal order of convergence by successively halving the iteration time-step. For this purpose the L2 norm of the error in density was computed. Table 5.1 shows that the Implicit BDF2 exhibits second order accuracy in time.

Unlike with the Backward Euler case, the number of symmetric forward-backward iteration sweeps and the frequency of gradient matrix computation can have a significant effect on the temporal accuracy. It was important to find the minimum number of forward-backward sweeps required before we could safely assume that the results are time accurate. For this purpose we looked at the deviation in the L2-norm of density error for different number of sweeps per time-step. The L2-norm obtained with 20 sweeps was assumed to correspond to exact time accurate solution, and the deviation was normalized with this exact value. Figure 5.10 shows a plot of the normalized deviation in the L2 error norm of density. It is observed that beyond 15 sweeps, the relative change in the L2 error norm is very small (of the order of 10^{-6}). It was also found that with 15 forward-backward sweeps, it was not necessary to compute the element gradient matrix each time-step. Reducing the frequency of gradient computation to upto once in 20 time-steps did not change the results noticeably.

Also, each implicit BDF2 iteration requires the solution at the n^{th} and $(n-1)^{th}$ time-step. Therefore there is a need for special treatment at the first iteration (t=0). In the present case, we run a few iterations of explicit RK5 to compute the solution at $t = \Delta t$. The solution at t=0 and $t = \Delta t$ can now be used as starting points (Q^n and Q^{n-1}) for the Implicit BDF2 algorithm.

Unsteady flow past plunging airfoil

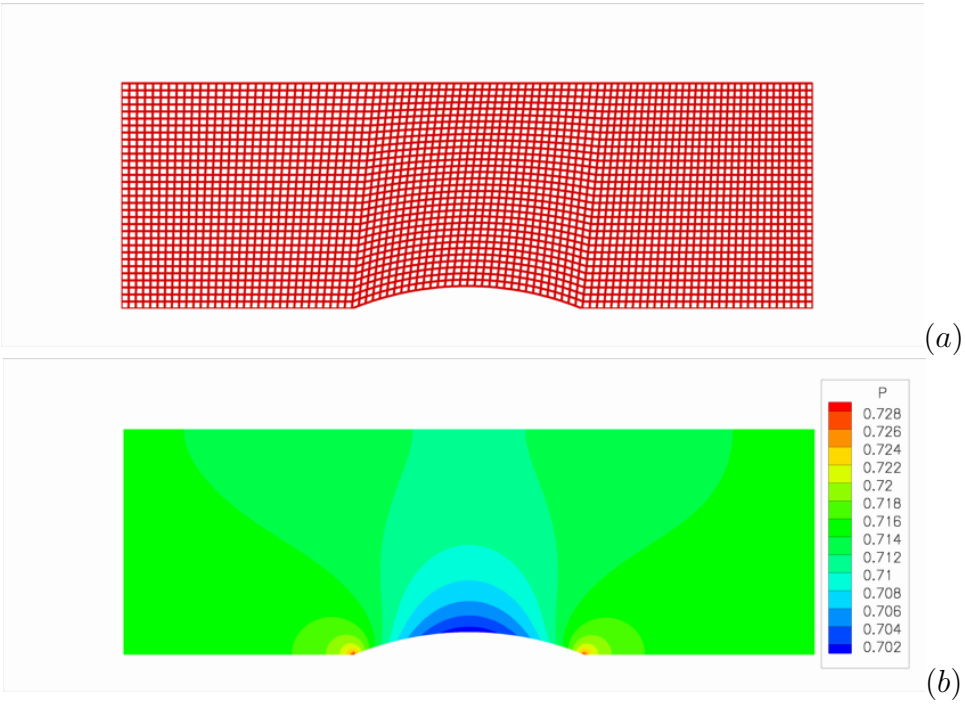
The current test case demonstrates the applicability of Implicit BDF2 time-stepping for unsteady flow problems. It also illustrates the good performance of high order methods for vortex dominated flows. The test case involves subsonic flow past a NACA0012 airfoil undergoing sinusoidal plunging motion. Following notations of Jones et al. [59], we define plunge amplitude as h and plunge circular frequency as $\omega = 2\pi f$. Subsequently, Strouhal number is determined as $Sr = \frac{\omega hc}{U_\infty}$, where c is the airfoil chord length and U_∞ is free-stream velocity. The airfoil plunge motion profile is prescribed as $Y(t) = h\cos(t)$. In the present case, we use $h = 0.12c$ and $\omega = 2.46$. The free stream Mach number is taken to be 0.2. A 5th order SD simulation is performed with Implicit BDF2 for time-stepping. The computational mesh used has 9984 cells with 84 nodes on the airfoil surface.

The results obtained compare very well with the experimental results for Jones et al [59] conducted in a water tunnel. The SD method is able to reproduce the primary wake - a dual-mode vortex street moving upwards at an angle to the airfoil. In addition, it is also able to capture the fine structures that are moving in a direction opposite to the primary wake. These flow features compare quite well with experiment (see Figures 5.16 (a) and (b)). Figures 5.11- 5.15 show the transient vortex interactions before the flow becomes periodic with a frequency equal to that of the plunging motion. These plots clearly indicate the detail with which the vortex interactions are captured, and goes to show the benefits of using high order spatial accuracy for such flows. It should be mentioned that lower order computations on a finer mesh (with comparable DOF's), were unable to resolve the fine structures moving downwards.

It was observed that the direction of the primary wake depended on the direction of the first stroke. If the first stroke was downwards, the dual-mode vortex street traveled upwards, and vice versa. This can be explained on the basis of the interaction between the leading edge vortex and the wake from the trailing edge. During each downward motion, a vortex separates from the LE and travels over the upper surface to the trailing edge, thus interacting with the wake there. This interaction between

the LE vortex over the upper surface, and the TE wake is the main reason behind the asymmetric flow pattern. Also, the surface over which the LE vortex moves is decided by the first stroke. If the first stroke is downwards, the LE edge vortex moves over the upper surface and tends to pull the primary wake upwards.

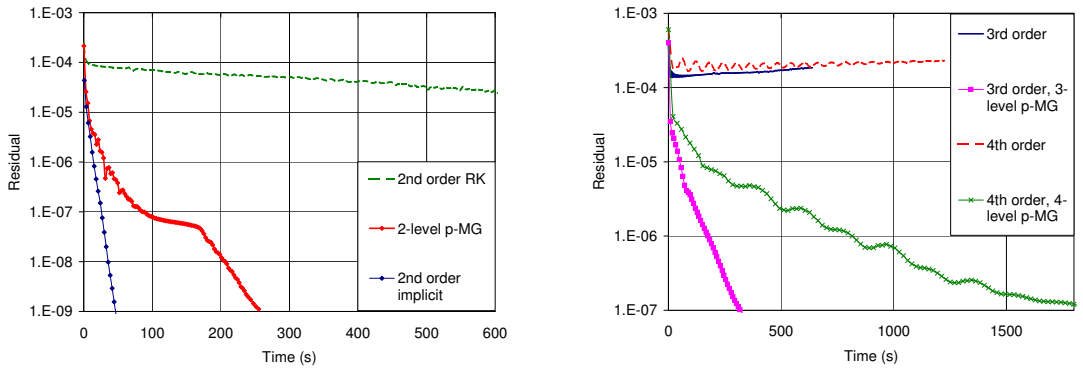
The lift and drag coefficients predicted by the 5th order SD method are shown in figure 5.17. The predicted time-averaged lift and drag coefficients are 2.58 and -0.51 respectively. It is clear that the frequency of the lift curve is equivalent to the plunge oscillating frequency. We compare computed lift and drag coefficients obtained by our SD viscous code with the ones obtained by the inviscid Panel code employed in (Jones et al., 1998). Both codes compute the same oscillating plunge motion. The difference in the lift coefficients predicted by both codes are very small. This can be attributed to the fact that lift due to pressure dominates the lift due to viscous stresses. However, viscous stresses play a more significant role in the streamwise direction. The resultant thrust force predicted by the SD code has a peak magnitude only 30% as big as the peak magnitude predicted by the Panel code.



(a)

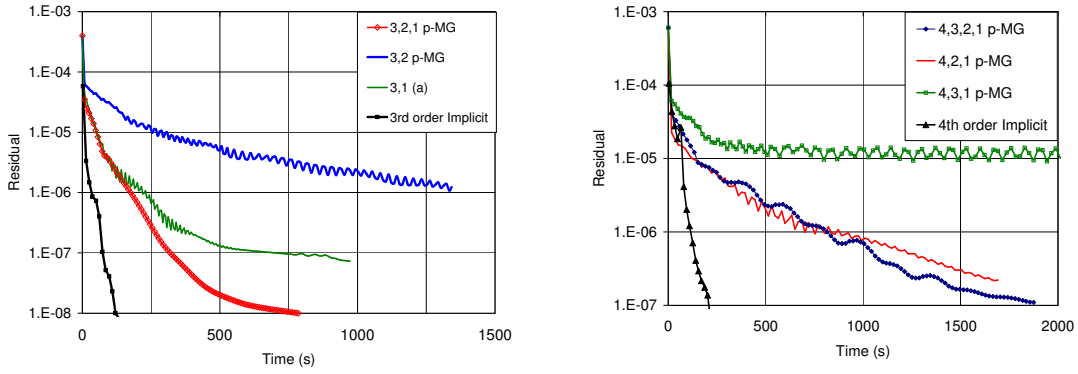
(b)

Figure 5.1: Inviscid subsonic flow past bump test-case (a) Computational grid (b) Pressure contours using 4th order SD



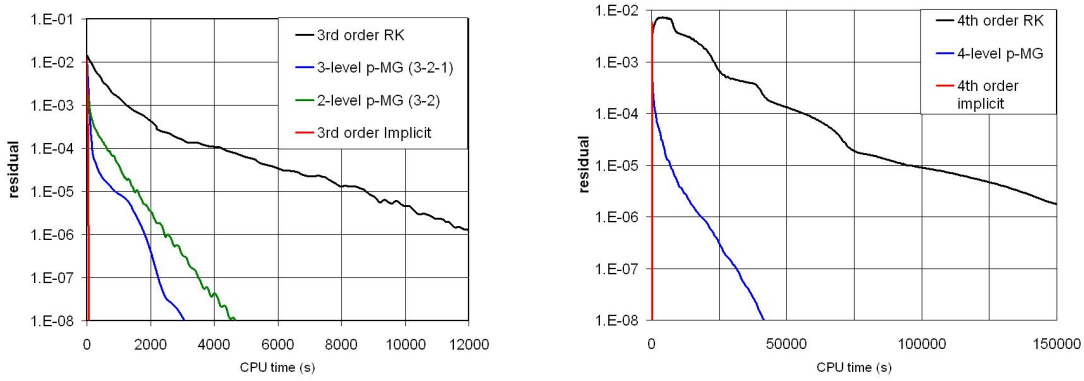
(a) (b)

Figure 5.2: Convergence history for subsonic flow over bump testcase (a) Plot of residual vs CPU time for 2nd order computations; (b) Plot of residual vs. CPU time for 3rd and 4th order computations



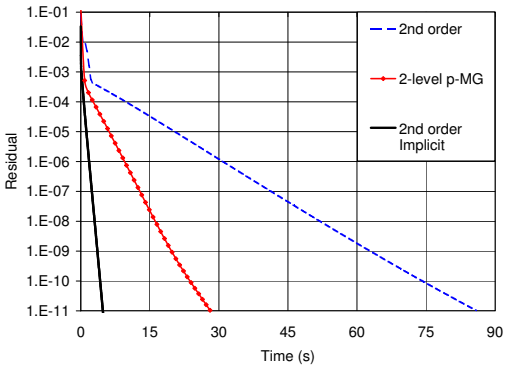
(a) (b)

Figure 5.3: Convergence history for subsonic flow over bump testcase (a) Plot of residual vs CPU time for 3rd order computations; (b) Plot of residual vs. CPU time for 4th order computations.

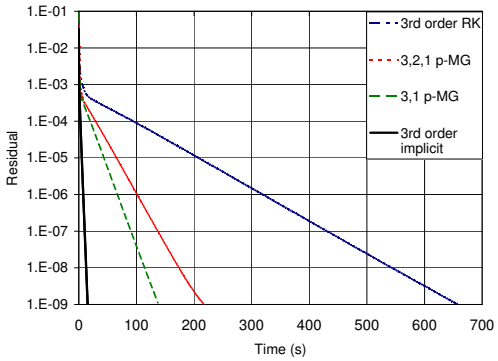


(a) (b)

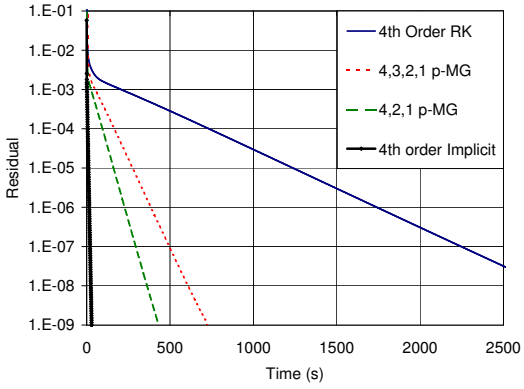
Figure 5.4: Convergence history for inviscid flow past NACA0012. (a) Plot of residual vs CPU time for 3rd order computations; (b) Plot of residual vs. CPU time for 4th order computations.



(a)



(b)



(c)

Figure 5.5: Convergence history Planar Couette flow case (a) 2nd order SD (b) 3rd order SD (c) 4th order SD

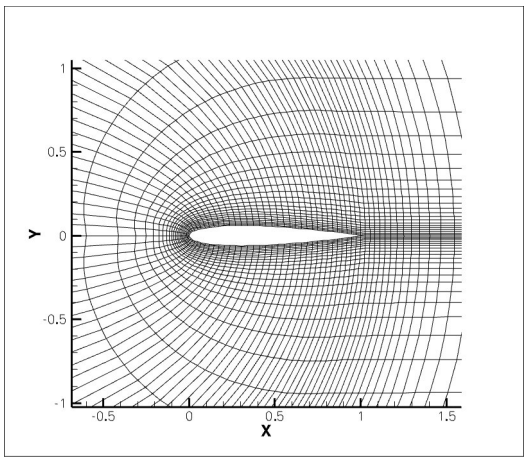
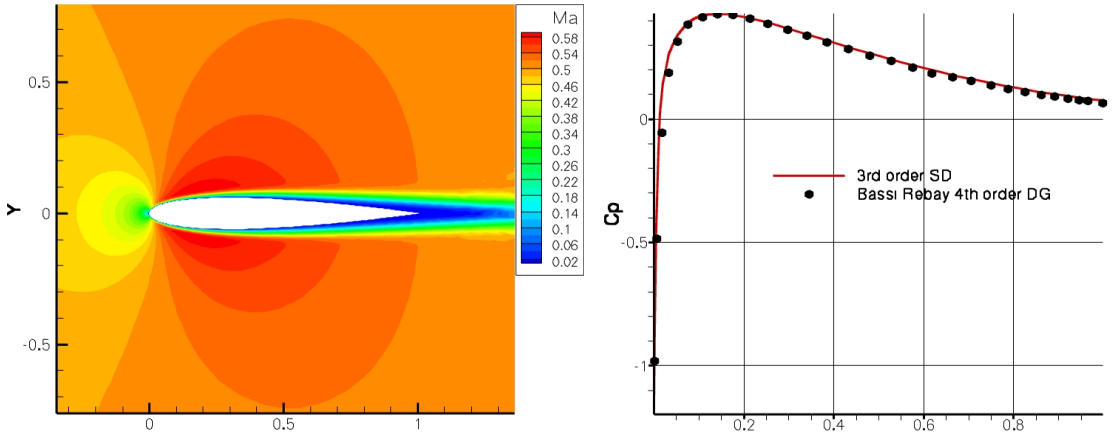
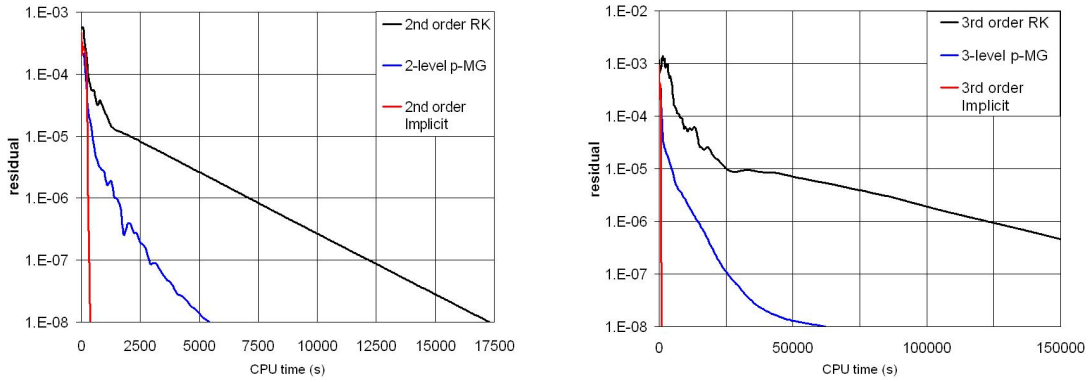


Figure 5.6: 120 × 24 C-grid for NACA0012 airfoil



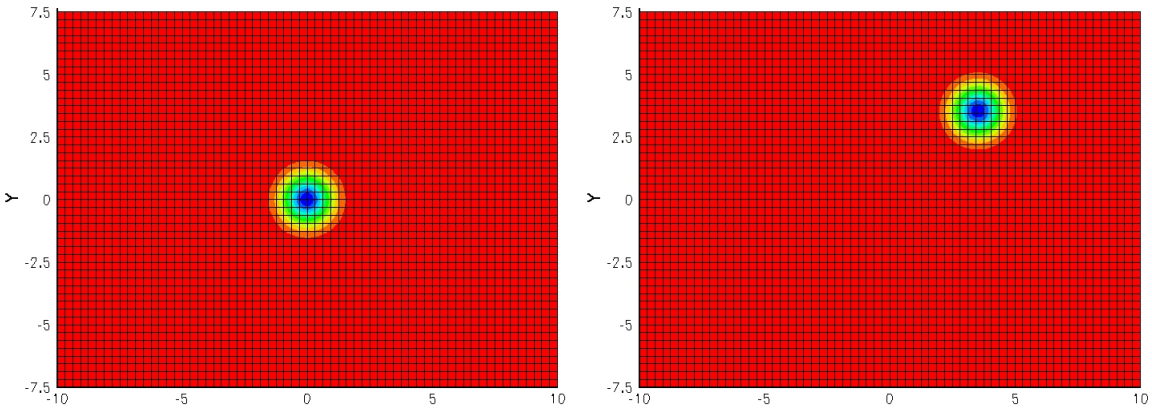
(a) (b)

Figure 5.7: Viscous flow around NACA0012 (a) Mach number contours using 3rd order SD; (b) Surface C_p distribution.



(a) (b)

Figure 5.8: Convergence history for viscous flow past NACA0012. (a) Plot of residual vs CPU time for 2nd order computations; (b) Plot of residual vs. CPU time for 3rd order computations.



(a) (b)

Figure 5.9: Density contours for euler vortex case. (a) at t=0 s (b) at t=10 s.

dt	L2-error	Temporal Order
0.4	2.192E-04	-
0.2	5.413E-04	2.02
0.1	1.430E-04	1.92
0.05	3.644E-05	1.97

Table 5.1: L^2 errors and orders of temporal accuracy for Implicit BDF2 with Euler-vortex test-case

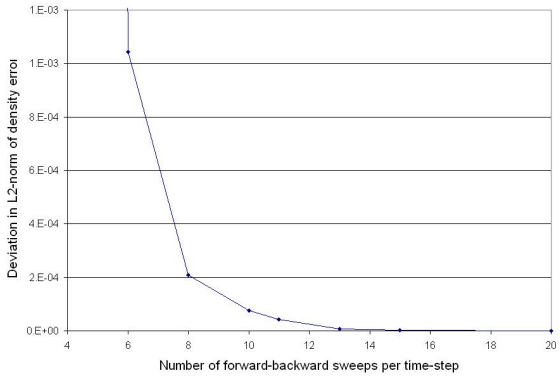
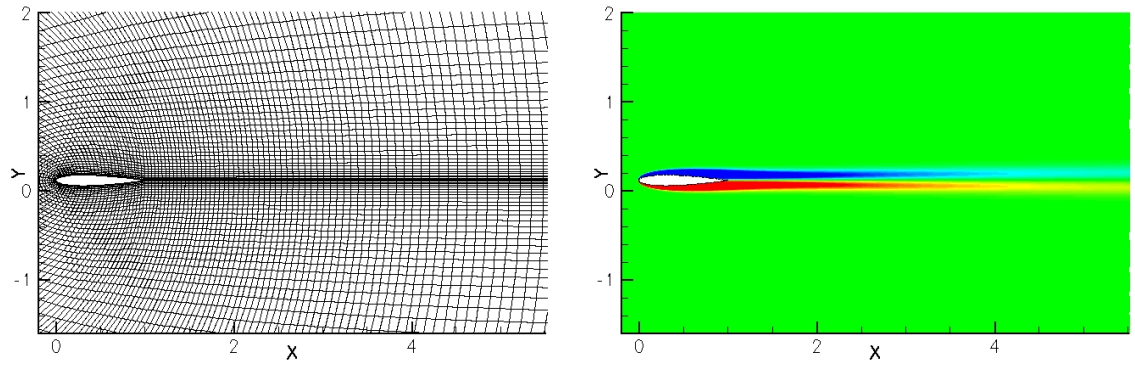
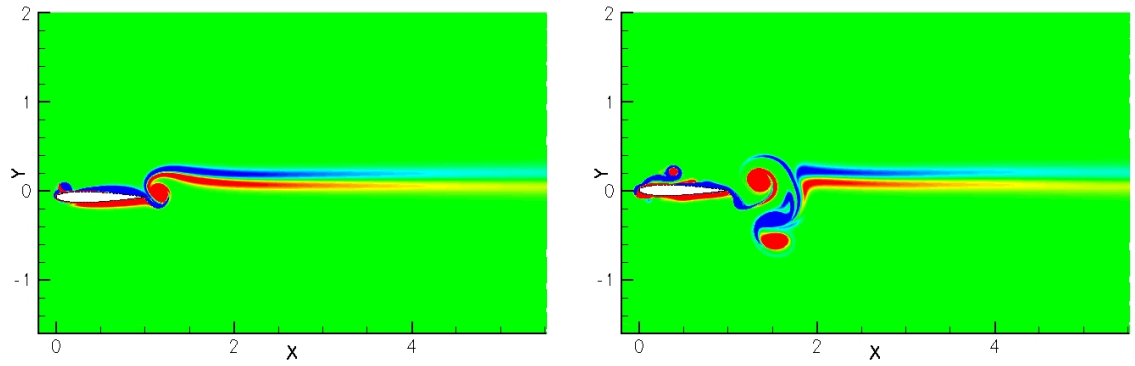


Figure 5.10: Plot of deviation in L2-norm of density error vs. number of sweeps per time-step, for the Euler vortex test case. The deviation has been normalized with the L2-norm value corresponding to 20 sweeps



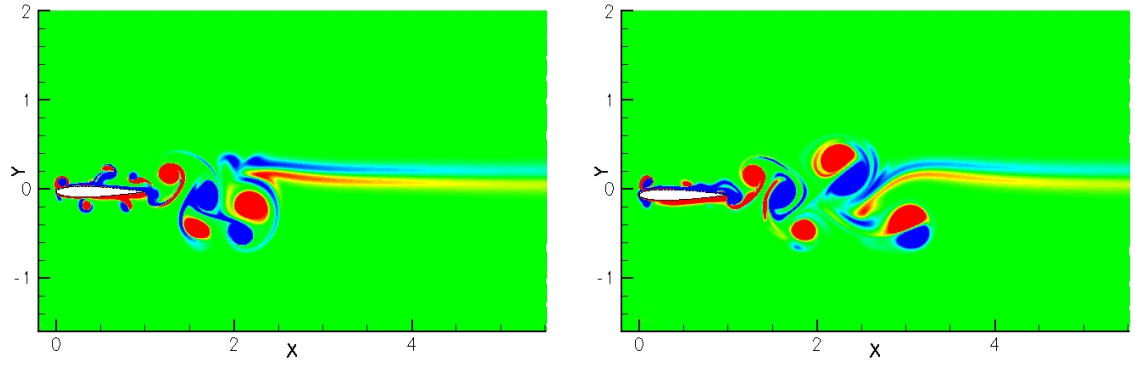
(a) (b)

Figure 5.11: Plunging airfoil test case (a) Mesh near the airfoil (b) Vorticity contours at t=0, 40 equally spaced contours between -1 and 1.



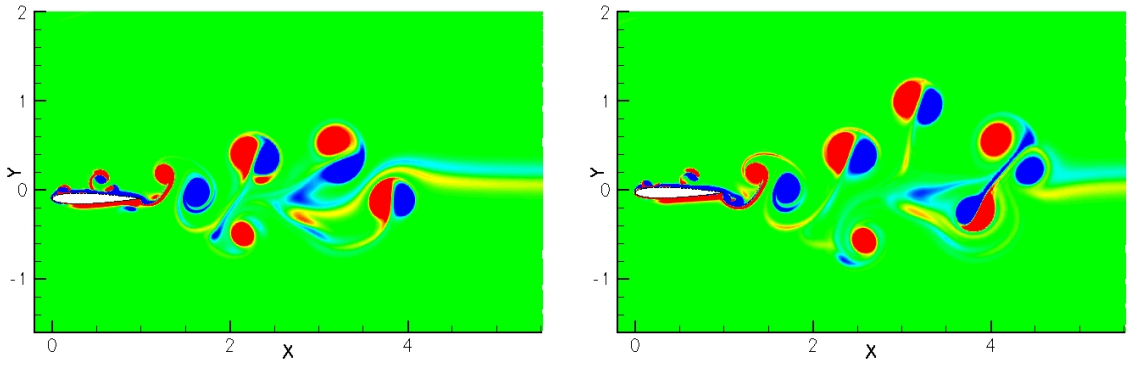
(a) (b)

Figure 5.12: Vorticity contours for plunging airfoil (a) at t=1.78 s (b) at t=4.29 s



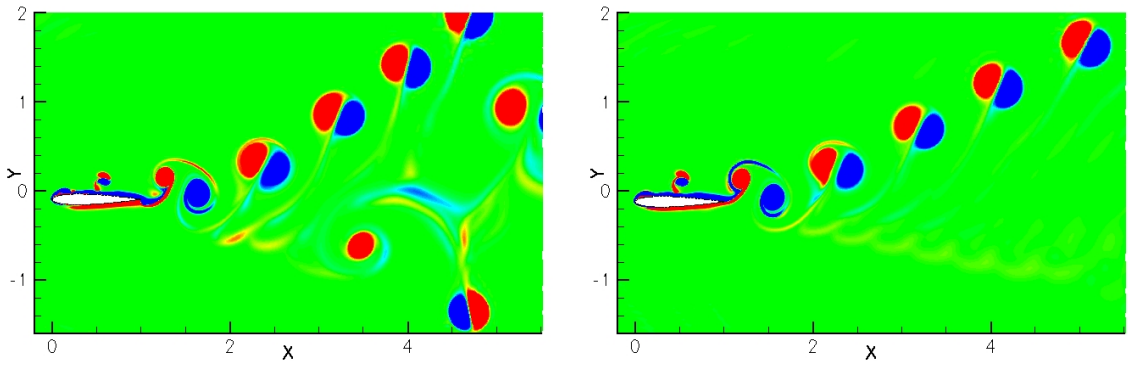
(a) (b)

Figure 5.13: Vorticity contours for plunging airfoil (a) at t=6.82 s (b) at t=9.34 s



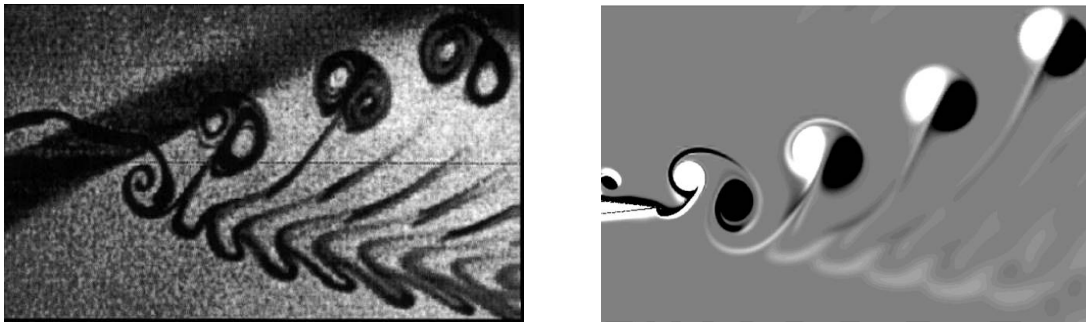
(a) (b)

Figure 5.14: Vorticity contours for plunging airfoil (a) at t=11.87 s (b) at t=14.39 s



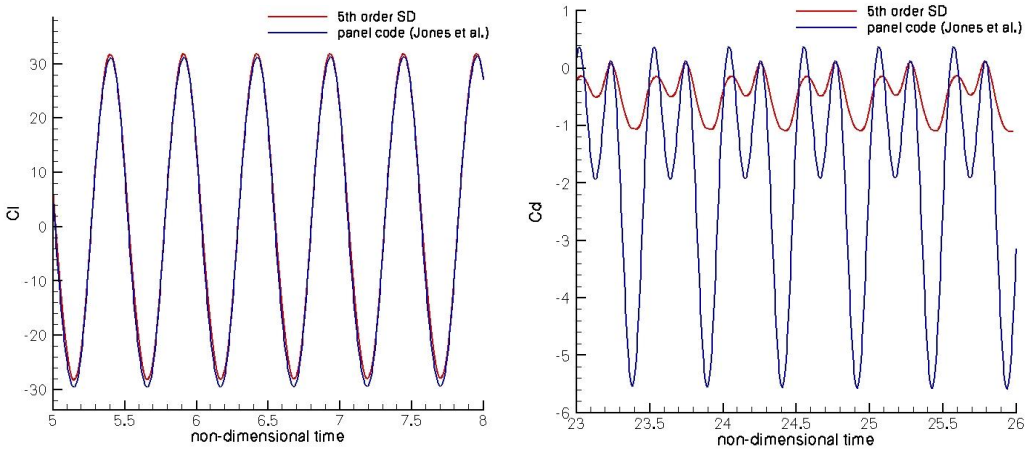
(a) (b)

Figure 5.15: Vorticity contours for plunging airfoil (a) at t=19.43 s (b) at t=39.6 s



(a) (b)

Figure 5.16: (a)Experimental results from Jones et al. [59] (b) Vorticity contours using 5th order SD



(a) (b)

Figure 5.17: Flow past plunging airfoil. Comparison of lift and drag with results obtained using panel code by Jones et al. (a) Lift Coefficient (b) Drag Coefficient.

Chapter 6

Shock Capturing using Artificial Viscosity

The application of spatially high-order schemes to compute flows with discontinuities such as shocks or contact-discontinuities, results in non-physical spurious oscillations that may cause the computation to go unstable. One of the major concerns in simulating such flows is to ensure the removal of these non-physical oscillations while avoiding damping of relevant flow features, such as resolved scales of turbulence. The artificial viscosity (AV) approach involves the explicit addition of dissipation in the region of flow discontinuities, thus smearing the discontinuity over resolution length scales, such that they can be well represented.

The present work proposes to add artificial grid-based components to the transport coefficients such as shear viscosity, bulk viscosity and the thermal conductivity. Such a form of artificial viscosity was introduced by Cook and Cabot for high-order centered differencing schemes, wherein a spectral-like high-wavenumber biased artificial viscosity and diffusivity were dynamically added [27, 26]. Further efforts by Fiorina and Lele [34], and Kawai and Lele [63] extended this high wave-number biased artificial viscosity (hyper-viscosity) approach to the Compact Difference Schemes. This approach was found to perform well for problems with shock turbulence interaction on structured grids [64]. This motivated the investigation of such an approach in combination with the Spectral Difference method.

As part of the current work, we attempt to extend the high wave-number biased AV formulation to an unstructured setup. Also, the original formulation scales the artificial viscosity coefficients as high order derivatives (bi-laplacian) of the sensor quantities. We investigate scaling the artificial coefficients as the 2nd and 4th derivatives of the sensor, and also directly as the sensor quantities. The proposed approach is applied to shock problems in 1D and 2D.

6.1 Formulation

The general formulation for artificial viscosity approach used here is similar to the 'Local artificial viscosity and diffusivity' approach of Kawai and Lele [63], which extends the original high-wavenumber biased artificial viscosity approach introduced by Cook and Cabot [27] to anisotropic and curvilinear structured grids. This current work adds grid-dependent components to the viscosity coefficients.

$$\begin{aligned} \mu &= \mu_f + \mu_\Delta \\ \beta &= \beta_f + \beta_\Delta \\ \kappa &= \kappa_f + \kappa_\Delta \end{aligned} \tag{6.1}$$

where μ is the dynamic (shear) viscosity, β is the bulk viscosity, and κ is the thermal conductivity. The f and Δ subscripts denote the fluid and artificial transport coefficients respectively.

These artificial transport coefficients are defined by:

$$\begin{aligned} \mu_\Delta &= C_\mu \rho \, | \sum_{l=1}^{2} \sum_{m=1}^{2} \Delta_l^{r+2} \left(\frac{\partial \xi_l}{\partial x_m} \right)^r \frac{\partial^r S}{\partial \xi_l^r} \, | \\ \beta_\Delta &= C_\beta \rho \, | \sum_{l=1}^{2} \sum_{m=1}^{2} \Delta_l^{r+2} \left(\frac{\partial \xi_l}{\partial x_m} \right)^r \frac{\partial^r (\nabla . \mathbf{u})}{\partial \xi_l^r} \, | \\ \kappa_\Delta &= C_\kappa \frac{\rho c_s}{T} \, | \sum_{l=1}^{2} \sum_{m=1}^{2} \Delta_l^{r+2} \left(\frac{\partial \xi_l}{\partial x_m} \right)^r \frac{\partial^r e}{\partial \xi_l^r} \, | \end{aligned} \tag{6.2}$$

where C_μ, C_β and C_κ are user-specified constants. ξ_l refers to the computational coordinates and x_m refer to the physical coordinates. Δ_l is the physical grid spacing

along a grid line in the ξ_l direction. The magnitude of the strain rate tensor (S), the dilatation ($\nabla.\mathbf{u}$) , and the internal energy (e) are the sensors corresponding to artificial shear viscosity, bulk viscosity and conductivity respectively. Experiments using high-order structured grid calculations suggest that r equals 4 [80]. For sufficiently high r, the high-wavenumber bias (k^r) results in damping of wavenumbers close to the unresolved wavenumbers.

It must be noted that such a formulation results in the addition of artificial viscosity terms that are $O(\Delta^{r+2})$ in smooth regions of the flow and $O(\Delta)$ in the vicinity of the shock. This can in fact be compared to the blended diffusion used by Jameson et al [53], in schemes like JST (Jameson-Schmidt-Turkel) and SLIP (Symmetric Limited Positive), where the artificial dissipation is third order in smooth regions of the flow and first order when there is a discontinuity.

Choice of parameter r

The value of parameter r is found to have a significant effect on the shock capturing properties of the present formulation. The value of this parameter is found to affect the smoothness of the artificial viscosity profile, the computational effort and the magnitude of AV in the smooth regions of the flow. The present work experiments with $r = 4, 2$ and 0. The main features of each choice are described below.

a) $r = 4$

This corresponds to computing the 4th derivatives of the sensor quantity. It also meant that the artificial coefficient is $O(\Delta^6)$ in smooth regions of the flow. However, the formulation with $r = 4$ presented a number of issues. Firstly, since we are using 3rd and 4th order accurate polynomial reconstructions within the cells, computing the 4th derivatives resulted in the introduction of numerical errors. This was found to severely degrade the solution quality in 2D when using non-cartesian meshes, and when the discontinuity is not aligned with the grid lines. Secondly there was significant computational effort involved. This is because we use locally discontinuous solution representations, and each level of differentiation involves extrapolation, averaging at interfaces and computing the derivative. Due to the aforementioned reasons,

we only present results for computations with $r = 2$ and $r = 0$.

b) r=2

Our earlier experiments with this form of viscosity used $r = 2$, which essentially meant adding a $O(\Delta^4)$ term to the viscosity coefficients in the smooth regions of the flows. It involves computing the 2nd derivatives of the sensor quantities. An advantage of such an artificial viscosity scheme is that it eliminates to an extent, the need for limiters/switches to turn off the artificial coefficient in smooth regions of the flow, such as regions of expansion and isentropic compression. However, the use of $r = 2$ was found to perform poorly when using curvilinear meshes, or fully unstructured meshes. This can be attributed to the fact that the calculation of the 2nd derivatives of the sensor quantities introduces errors, which are more severe when the shock is not aligned with the grid-lines.

c) $r = 0$

When $r = 0$ the general formulation for artificial viscosity reduces to

$$\begin{aligned} \mu_\Delta &= C_\mu \overline{\rho \mid \Delta^2 S \mid} \\ \beta_\Delta &= C_\beta \overline{\rho S_\beta \mid \Delta^2 \left(\nabla.\mathbf{u}\right) \mid} \end{aligned} \tag{6.3}$$

where Δ is the grid spacing, and S_β is a dilation-based switch.

This form of artificial viscosity was found to give much better results for flows with shocks on curvilinear and fully unstructured meshes. This is clearly demonstrated in figure 6.3. Figure 6.3(a) shows the computed artificial viscosity (with r=2) when a shock is located at x=0.0, and the discontinuity is aligned with the grid lines. However in figure 6.3(b), the discontinuity (located at x=0.5) is not aligned with the mesh lines. Here, with $r = 2$, we see non-smooth AV contours and non-physical transverse variations in AV. However, while using r=0 for the AV computations, we see that the AV profile is much smoother, and there are no significant transverse variations, as shown in figure 6.3(c). The superior performance of using r=0 is also seen when using a fully unstructured grid as shown later in Section 6.2.

However reducing r to 0 comes with certain disadvantages. Firstly the artificial

viscosity term becomes 2nd order $O(\Delta^2)$ in smooth regions, which is obviously less accurate than the higher order terms obtained with $r \geq 2$. Secondly, the formula for the artificial conductivity becomes invalid. It is however possible to find an alternate formulation for problems with contact discontinuities. Thirdly, there is now a need for a switch (S_β in the formulation) to turn off artificial viscosity in regions of smooth flow (which is explained later). The one advantage though is that the computational cost for calculating the artificial viscosity becomes much lesser.

Filter for unstructured SD setup

The overbar in equation 6.3 denotes a filter to smooth the artificial transport coefficients. The filter is meant to eliminate cusps introduced by the absolute value operator, which in turn ensures that artificial viscosities are positive. The filter plays an important role in artificial viscosity computations as it ensures smooth variation of artificial transport coefficients within the domain. For calculations using artificial viscosity on structured grids, a truncated Gaussian filter is used. A 7-point or 9-point stencil is generally used for this purpose [27]. For calculations in 2D, the Gaussian filter is applied along each grid-line separately. However, for unstructured grids it is not reasonable to implement the Gaussian filter in its existing form, as obtaining a stencil for each solution/flux point can be tedious. The stencil would lie across cells, and the non-uniform spacing would have to be taken into account, thus making it cumbersome to implement. This motivated the development of a filter that would be suited to the current SD setup.

In the current study we use an element-wise restriction-prolongation filter (we will refer to it as the R-P filter). The concept is similar to the one used by Blackburn et al [18] for spectral element filtering. It involves the projection of the quantity in concern to a lower-order basis (restriction), smoothing at this level, and then extrapolation back to original basis (prolongation). The basic steps in implementation of the R-P filter can be described in 1-D as follows,

1. Consider a 4th order SD element. The artificial viscosity terms have been computed at the 4 solution points (Figure 6.1(a)).

2. The function (represented by a cubic polynomial through the 4 solution points) is restricted to 2 solution points (corresponding to 2nd order SD) (Figure 6.1(a)). The polynomial fit through the interpolated function is reduced to linear. The function is now extrapolated to the 3 flux points corresponding to a second order solution.

3. The function values are averaged at the interface for all element interfaces (Figure 6.1(c)). This is equivalent to smoothing of the function at the lowest level. A quadratic polynomial is fitted through this smoothed function through the 3 flux points.

4. It is then extrapolated to the flux points at the highest level (4th order)(see Figure 6.1(d)).

Figure 6.2(a) corresponds to the initial condition of the SOD shock tube case with a density discontinuity at x=0.5. The artificial conductivity is non-zero in the vicinity of x=0.5. The artificial coefficients were computed using r=2. The figure shows that prior to filtering the artificial conductivity field is noisy and may have oscillatory behavior. The filtered coefficient is smoother and results in a better solution. Figure 6.2(b) shows the density jump for the advancing shock front at time $\tau = 0.15$. It is clear that non-smoothed AV profile is not as effective in eliminating the non-spurious oscillations. Also, in 1-D, this filter performs comparably to the Gaussian filter applied on flux points. It must be mentioned that the effect of the filter is more significant for higher values of r. Since the solution representation in multiple dimensions is just a tensor product of 1-D polynomials, the extension of this filter to 2-D and 3-D is straight-forward.

Switch in artificial viscosity formulation

For the AV formulation with r=0, the artificial coefficients are $O(\Delta^2)$ in smooth regions of the flow. This can seriously contaminate the solution, especially for 3rd and 4th order accurate computations. Hence we need a switch to turn off AV in smooth regions of the flow. The switch used here is one proposed by Bhagatwala and Lele [17] to restrict the addition of AV only to regions of strong shocks. The formulation

of the switch is as shown

$$S_\beta = 0.5 * \left(1 - tanh(C1 + C2 * \frac{\Delta}{c} \nabla.\mathbf{u})\right) \tag{6.4}$$

The constants C1 and C2 were chosen to be 2 and 20 respectively for the present cases. This switch is based on the value of dilatation and is designed to add AV only in regions of strong negative dilatation, i.e, corresponding to shocks.

Steps involved in computation of AV coefficients

The artificial viscosity/conductivity is calculated at each flux point. This is computationally extensive, but it ensures smooth variation of artificial transport coefficients. A smooth representation of artificial viscosity within mesh elements is considered beneficial as compared to the piecewise-constant artificial viscosity formulations, as element-to-element variations can lead to oscillations in state gradients and disparate equilibrium shock-jump conditions in neighboring elements [7].

The implementation of the artificial viscosity method to the SD solver can be summarized in the following steps

1. The sensor is computed at the solution points.

2. For $r \geq 0$, the sensor is then extrapolated to the flux points, and differentiated. This step can be repeated required number of times to obtain $\frac{\partial^r S}{\partial \xi_l^r}$ at the solution points.

3. Similarly, the partial derivatives of the other sensors can also be computed, if necessary.

4. The artificial transport coefficients are computed at the solution points using equations 6.2 or 6.3.

5. The coefficients are then filtered and reconstructed to the flux points as described earlier.

6.2 Results

SOD Shock-tube problem

The first 1-D test case is the shock-tube problem introduced by SOD [107]. The initial left and right-side conditions are $\rho_l = 1.0$, $u_l = 0.0$ and $p_l = 1.0$ for $x \leq 0.5$, and $\rho_r = 0.125$, $u_r = 0.0$ and $p_r = 0.1$ for $x > 0.5$. Simulations are performed on a uniformly spaced grid in the region $0 \leq x \leq 1$. Artificial bulk viscosity and conductivity are used. $r = 2$ is used for this case. The user-defined coefficient values used are $C_\beta = 0.06$ and $C_\kappa = 0.01$ for the 4^{th} order computation. C_μ is set to zero.

Figure 6.4 (a),(b) and 6.5 (a) shows the comparison between density, velocity and pressure for the exact solution and 4th order SD computation with 100 cells at non-dimensional time $\tau = 0.15$. The shock and the contact discontinuity are captured well without significant spurious oscillations, and show reasonable agreement with the exact solution. Figure 6.5 (b) shows the variation of density profile with grid-refinement. It is observed that as the grid is refined, the solution converges closer to the exact solution. It should also be noted that for all grid spacings, the shock is spread over one-two cells and the contact discontinuity is spread over two-three cells. Figure 6.6 (a) shows the artificial bulk viscosity coefficient. It is seen to be maximum in the vicinity of the shock. Figure 6.6 (b) shows the artificial conductivity. There are two peaks corresponding to the shock and the contact discontinuity. In the SOD problem, artificial conductivity plays an important role because the artificial viscosity sensor does not sense the contact discontinuity.

The results described above were obtained using $r = 2$ in the artificial transport coefficient calculations. Computations were also conducted using $r = 4$, but the results obtained were very similar to those obtained for $r = 2$. For $r = 4$, $C_\beta = 0.001$ and $C_\kappa = 0.0001$ were used. For both 1-D cases it was observed that $r = 2$ and $r = 4$ gave very similar results.

Shu-Osher problem

The second 1-D test case is the shock-entropy wave interaction introduced by Shu and Osher [106]. The entropy waves are sensitive to numerical dissipation, excessive

numerical dissipation can damp the entropy waves. Initial left and right side conditions are given by: $\rho_l = 3.857143$, $u_l = 2.629369$ and $p_l = 10.33333$ for $x < -4$, and $\rho_r = 1 + 0.2 * sin(5x)$, $u_r = 0.0$ and $p_r = 1.0$ for $x \geq -4$. Simulations are performed on a uniformly spaced grid in the region $-5 \leq x \leq 5$. The coefficients for the 4^{th} order simulation are the same as for the SOD case.

Figure 6.7(a) shows the comparison between the reference solution and 4th order SD simulations with 100, 200 and 400 cells. The reference solution is obtained using 5th order WENO on 2000 grid points. The density profile using 400 cells shows excellent agreement with the reference solution. Figure 6.7(b) shows a close up of the density plot in the region of the entropy waves. It is found that solution with 200 cells also shows reasonable agreement with reference solution. Figures 6.8(a) and (b) show the velocity and pressure profiles behind the shock.

Stationary shock in 1D test-case

This 1D test case corresponds to stationary normal shock at Mach number 3.0. The initial conditions for the flow are prescribed using the Rankine-Hugoniot relations for a stationary shock. The computational domain has size 1 unit and the shock is located at x=0.5. For this case, we used r=0 with $C_\beta = 1.2$. Third order computations were done with 50, 100, 200 and 400 cells. The stationary shock is captured well without significant oscillations or overshoot (see Figure 6.9(a)). As the mesh is refined, the shock profile becomes sharper and the result converges to the exact solution. We also see that the shock profiles have a very small overshoot. Figure 6.9(b) plots the percentage overshoot (normalized by the shock pressure jump) versus the grid spacing. It is clear that as the grid spacing decreases, the percentage overshoot decreases.

Inviscid Supersonic flow past bump

This test-case consists of inviscid supersonic flow in a channel with a 4% thick circular bump on the bottom. The length of the channel is 3 units and its height 1 unit. The inlet Mach number is 1.4. This test case has been used by Ripley et al. [101] in computations using adaptive unstructured mesh refinement. Third and fourth order

SD computations were conducted on two meshes. The coarse computational mesh has 1200 elements, and 20 nodes to resolve the bump, as depicted in figure 6.10. The fine mesh has 4800 cells, and has twice the number of nodes in the x and y directions. The surface of the bump is represented using quadratic and cubic boundary for third and fourth order calculations respectively.

The pressure contours obtained using the 3rd order SD scheme with artificial viscosity on the coarse mesh is shown in figure 6.12(a), and compares well with those obtained using adaptive unstructured mesh refinement [101]. The pressure contours obtained for the fine mesh are shown in figure 6.12(b). It is observed that on the finer mesh, the shock profiles are sharper, and smoother contours are obtained. Figure 6.14 shows the drop in global residual, indicating a stable, convergent solution for both 3rd and 4th order cases. It must be mentioned that in the absence of artificial viscosity, the solution develops spurious oscillations and the simulation becomes unstable.

Figure 6.12(c) gives a plot of the artificial viscosity for third order computation on the coarse mesh. Figure 6.12(d) shows the variation of artificial bulk viscosity on the fine mesh. We see that artificial viscosity is added only in regions corresponding to shocks. Figures 6.13(a) and (b) shows the pressure contours obtained using 4th order SD on the coarse and fine mesh. It is observed that the shock profiles obtained using 4th order are slightly sharper than in the case of the third order computation. As expected, the 4th order computation on the finer mesh gives sharper shock resolution and more accurate contours in comparison to the coarse mesh.

The above test-cases were computed using r=0 and $C_\beta = 0.3$ for 3^{rd} order and $C_\beta = 0.15$ for 4^{th} order computations. Also, the artificial shear viscosity and artificial conductivity were set to zero. This is because there are no large shear gradients, and no contact discontinuities, and hence artificial bulk viscosity is sufficient to stabilize the calculations.

We also compare the results obtained using $r = 0$ and $r = 2$ with an unstructured mesh for the same configuration. The mesh used is shown in figure 6.11. The pressure contours for the $r = 0$ case shown in figure 6.15(a) indicate the presence of spurious oscillations and erroneous behavior downstream of the shocks. Comparatively, much smoother contours and better shock resolution are obtained with $r = 0$ (see

figure 6.15(c)). Also, for the r=0 computation, the AV contours obtained are found to be much smoother as compared to the r=2 case (see figures 6.15(b) and (d)). This case clearly demonstrates the superior behavior obtained with $r = 0$ for non-cartesian meshes.

Inviscid Supersonic flow past circular cylinder

The next test-case is the $M_\infty = 3.0$ flow past a circular cylinder. The computational grid with 1200 cells is shown in figure 6.16(a). Supersonic inflow and outflow BC's are used. The cylinder wall is treated as inviscid curved wall. In the artificial viscosity model, r=0 was used with $C_\beta = 1.0$ for 3rd order computation and $C_\beta = 0.5$ for 4th order computation. Pressure and artificial bulk viscosity for the 3rd order simulation and 4th order simulation are shown in figure 6.16 and figure 6.17 respectively. It is observed that the shock profile is thinner for the 4th order case as compared to the 3rd order case. For this test-case, the computation with r=2 was unable to converge to steady state, with the solution developing oscillatory behavior away from the center-line where the mesh is not aligned with the shock.

Inviscid Transonic flow past airfoil

This test-case involves the transonic flow past a NACA0012 airfoil. The free-stream mach number is 0.8 and the airfoil is at an angle of attack of 1.25 degrees. Free-stream conditions are prescribed at the outer boundaries. On the airfoil surface, inviscid wall boundary condition is used. The curved airfoil surface is represented by cubic splines. A weak shock is formed on the upper surface, and an even weaker shock is formed on the lower surface. The current method is able to capture the shocks accurately for both 3rd and 4th order computations. Computations are done on a 80×16 mesh and on a 160×32 mesh.

Figures 6.18 and 6.19 show the pressure and AV contours for the 3rd order computations on the coarse and fine mesh. Figures 6.20 and 6.21 show the pressure and AV contours for the 4th order computations on the coarse and fine mesh. A look at the AV plot clearly indicates that AV is added only in the region of the shocks. The switch ensures that AV is zero in regions of smooth flow even when dilatation

is non-zero, such as in the regions near the leading edge and trailing edge. The C_p plots obtained are compared with the results from FLO-82 on a 320x64 grid, and are shown in figure 6.22. The C_p plots obtained shows good comparison with those obtained using FLO-82 with a finer (320x64) grid. It is observed that the shock profile obtained with FLO-82 is much sharper. This is expected as the SD computations were done on coarser grids (even though the number of DOF's are comparable), and the current AV model tends to smear the shock over one to two cells.

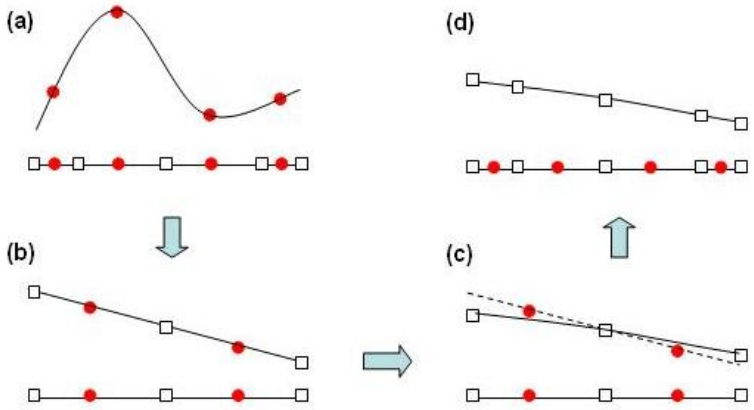


Figure 6.1: Steps involved in implementation of filter for 4th order SD

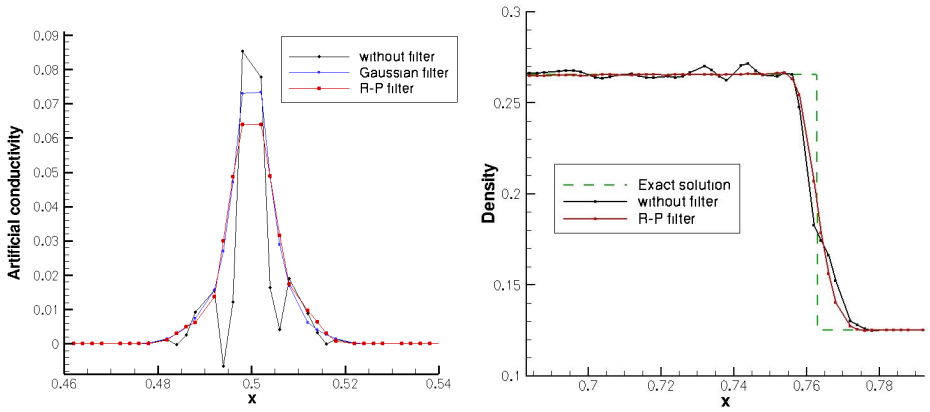


Figure 6.2: (a)Effect of filtering on the smoothness of artificial viscosity/conductivity coefficients (at $\tau = 0$) (b)Effect of filtering on density profile for shock (at $\tau = 0.15$)

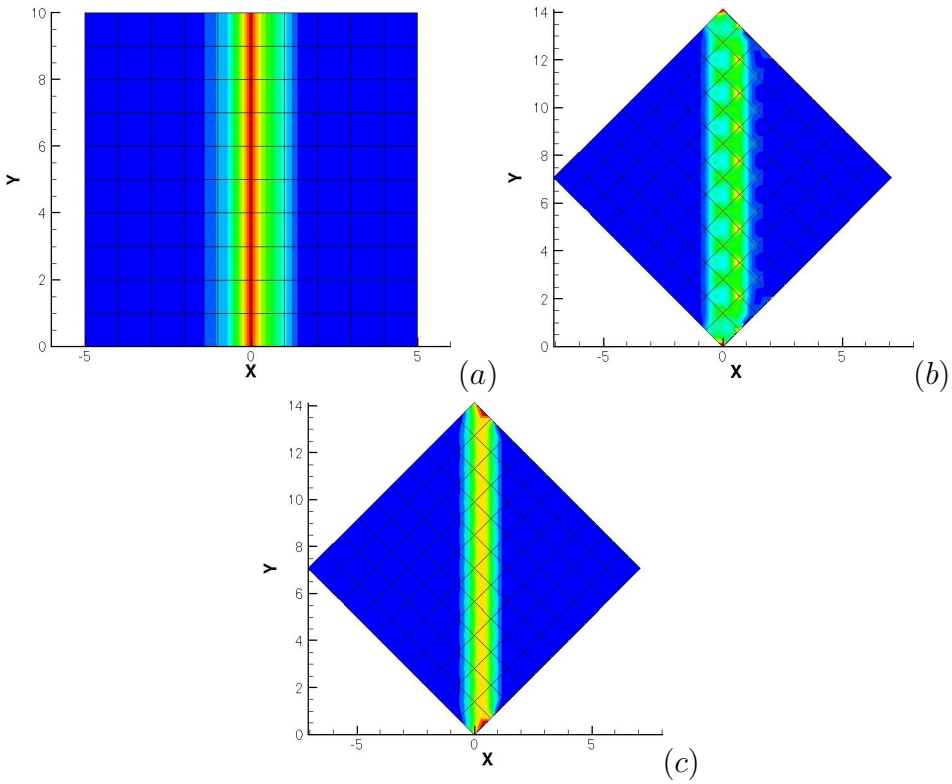
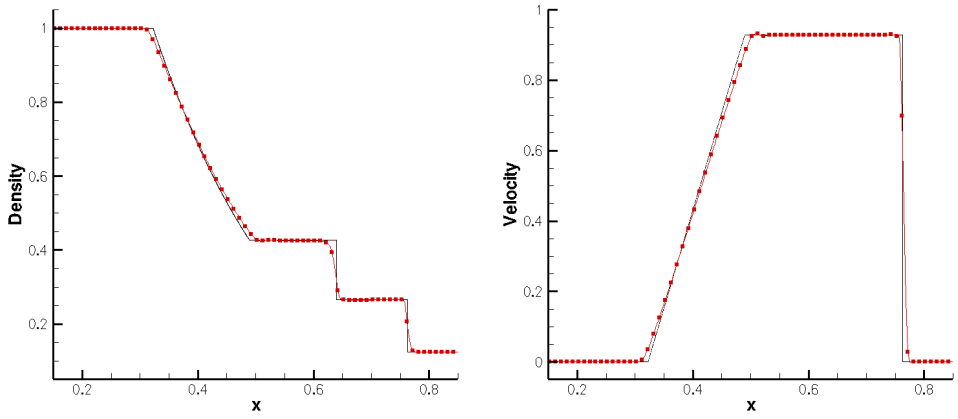
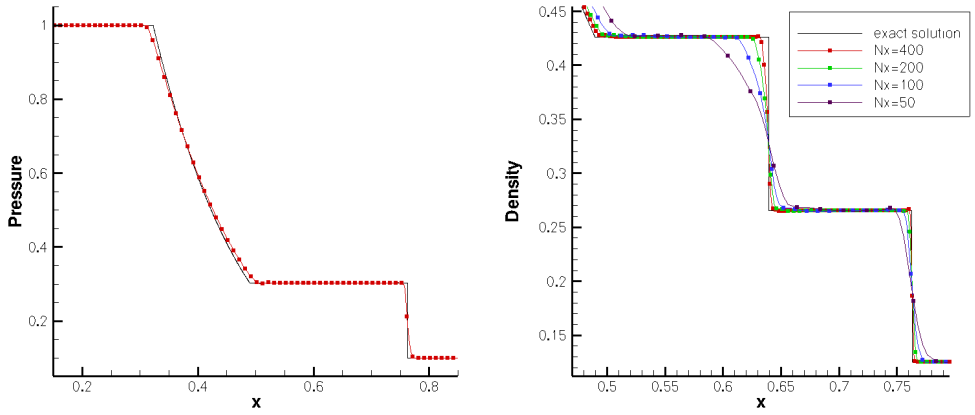


Figure 6.3: Artificial viscosity contours using (a) r=2 when discontinuity is aligned with grid lines (b) r=2 when discontinuity is not aligned with grid-lines (c) r=0 when discontinuity is not aligned with grid-lines



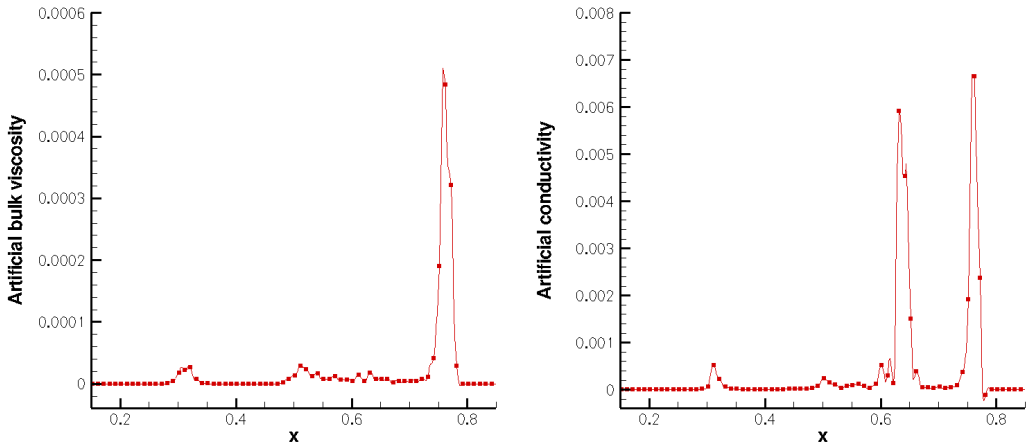
(a) (b)

Figure 6.4: SOD shock tube case, black line - exact solution, red line - 4th order SD with 100 cells at $\tau = 0.15$ (a) density vs. x ; (b) velocity vs. x.



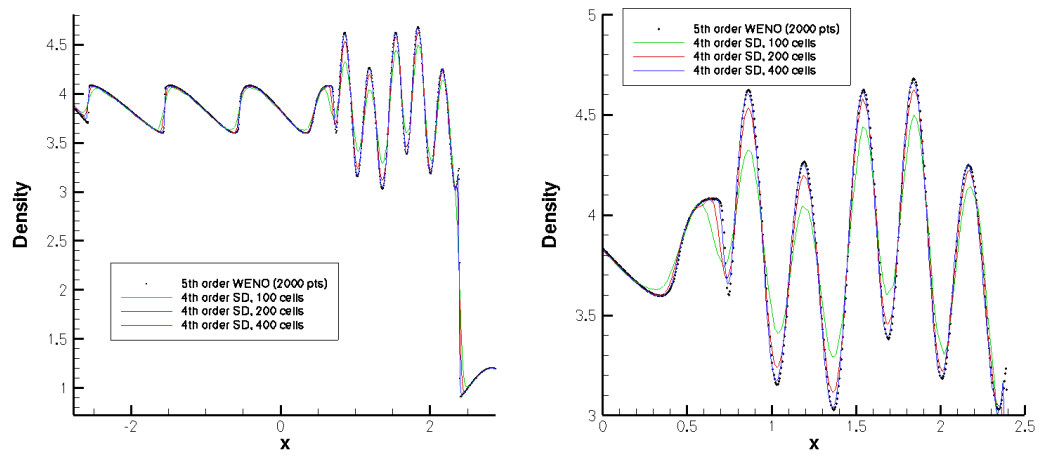
(a) (b)

Figure 6.5: SOD shock tube case, (c) pressure vs. x ; (d) effect of grid-refinement on density (Nx=number of cells).



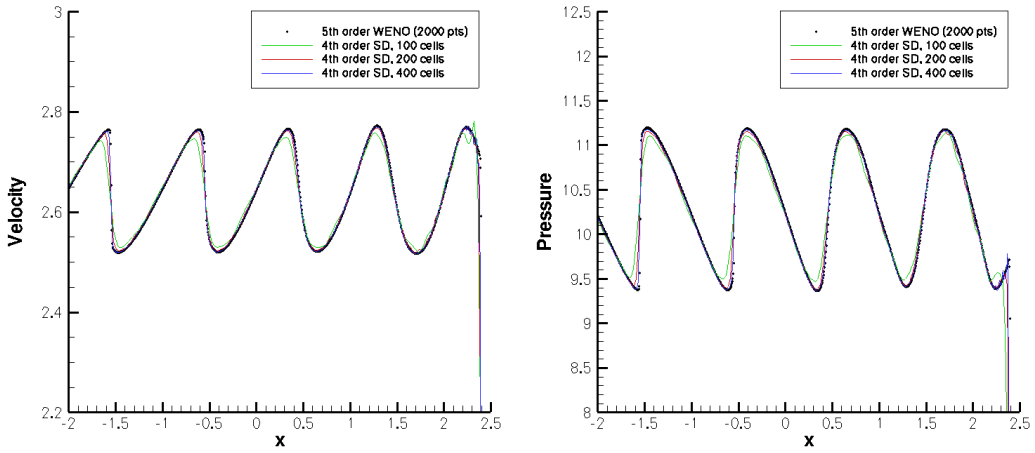
(a) (b)

Figure 6.6: SOD shock tube case (a) Artificial bulk viscosity; (b) Artificial conductivity at $\tau = 0.15$



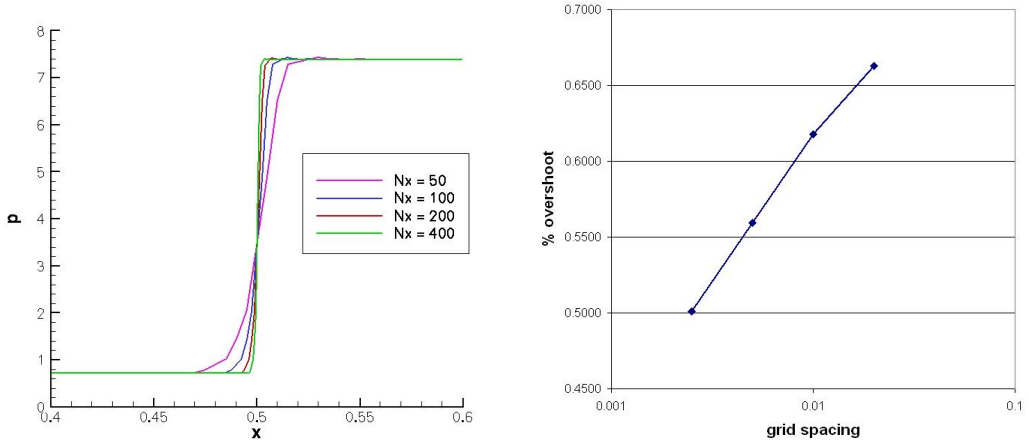
(a) (b)

Figure 6.7: Shu-Osher Shock turbulence interaction. Density is presented at $\tau = 1.8$.



(a) (b)

Figure 6.8: Shu-Osher Shock turbulence interaction. Plot of (a)Velocity, (b)Pressure, at $\tau = 1.8$.



(a) (b)

Figure 6.9: Stationary shock test case. (a) Pressure profiles for shock discontinuity (b) Variation of percentage overshoot with grid spacing

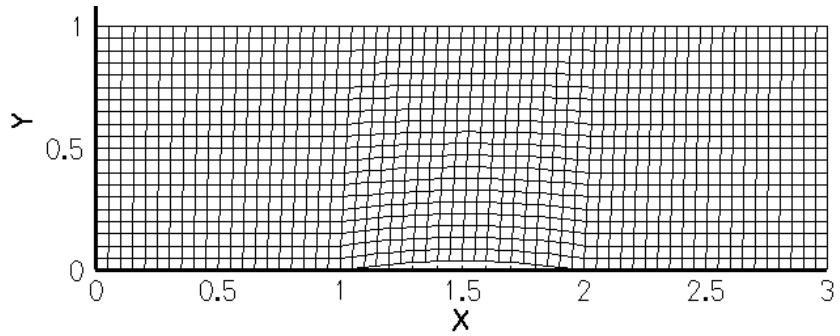


Figure 6.10: 60×20 computational grid for supersonic flow past bump

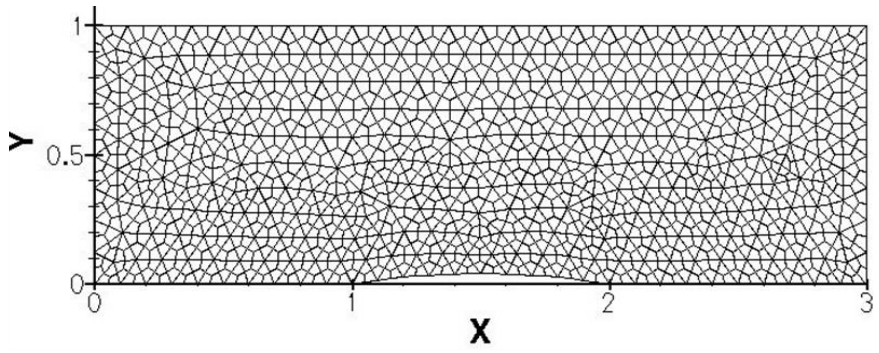


Figure 6.11: Fully unstructured grid for supersonic flow past bump

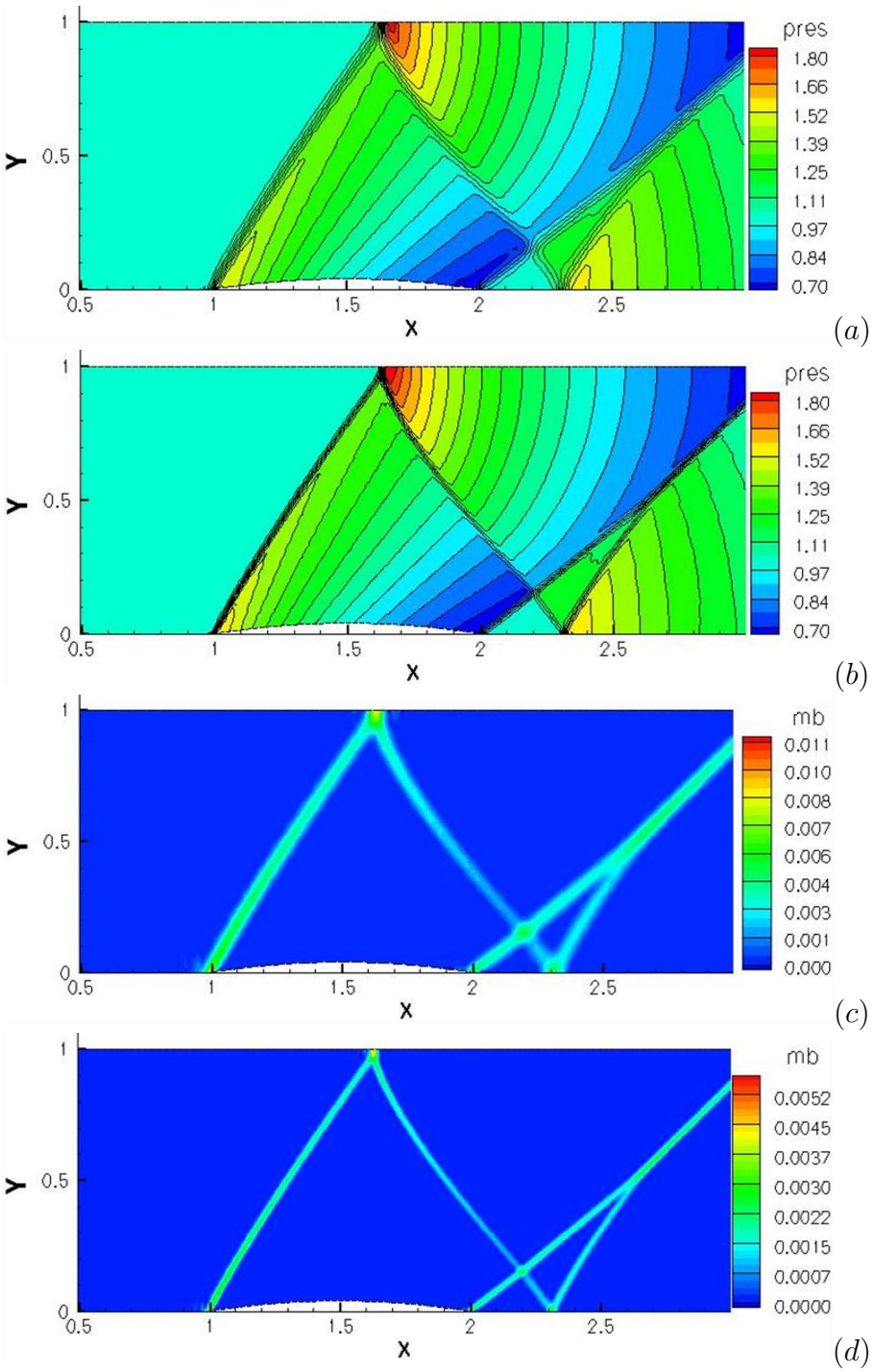


Figure 6.12: 3rd order SD computation with artificial viscosity. Non dimensional pressure contours for (a)60×20 mesh (b)120×40 mesh. Artificial bulk viscosity for (c)60×20 mesh (d)120×40 mesh.

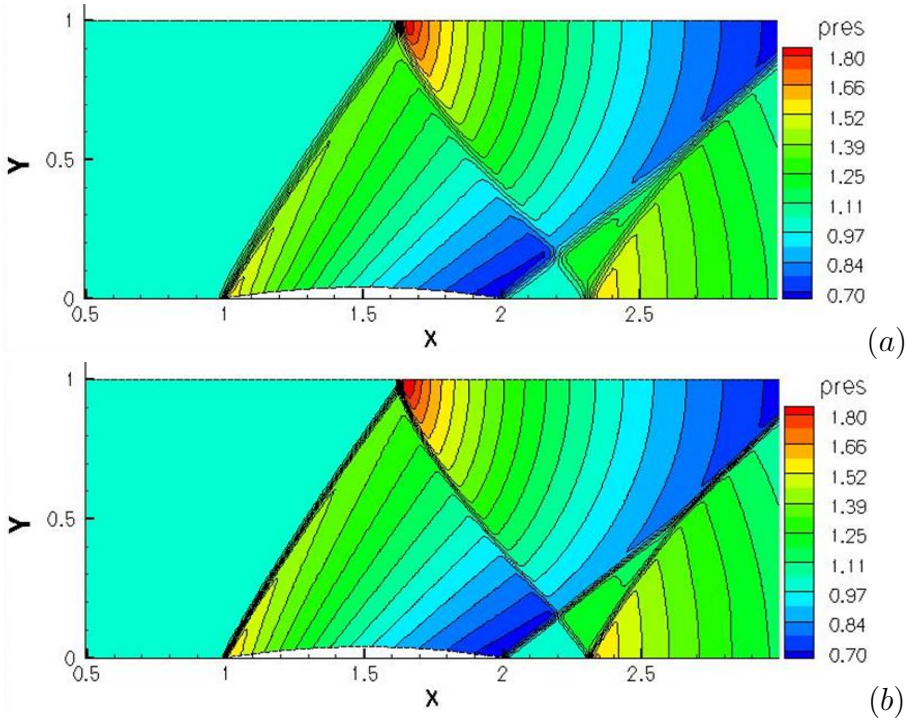


Figure 6.13: 4th order SD computation with artificial viscosity. Non dimensional pressure contours for (a)60×20 mesh (b)120×40 mesh.

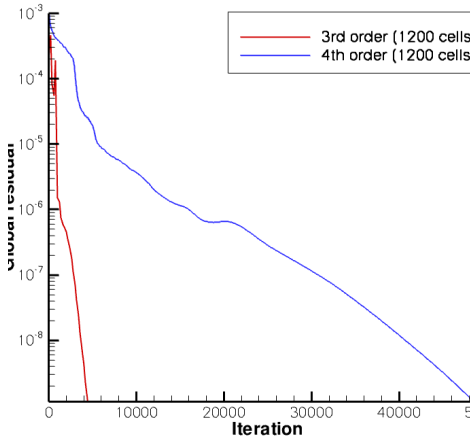


Figure 6.14: Convergence plot for supersonic bump flow case using 3rd and 4th order SD with artificial viscosity

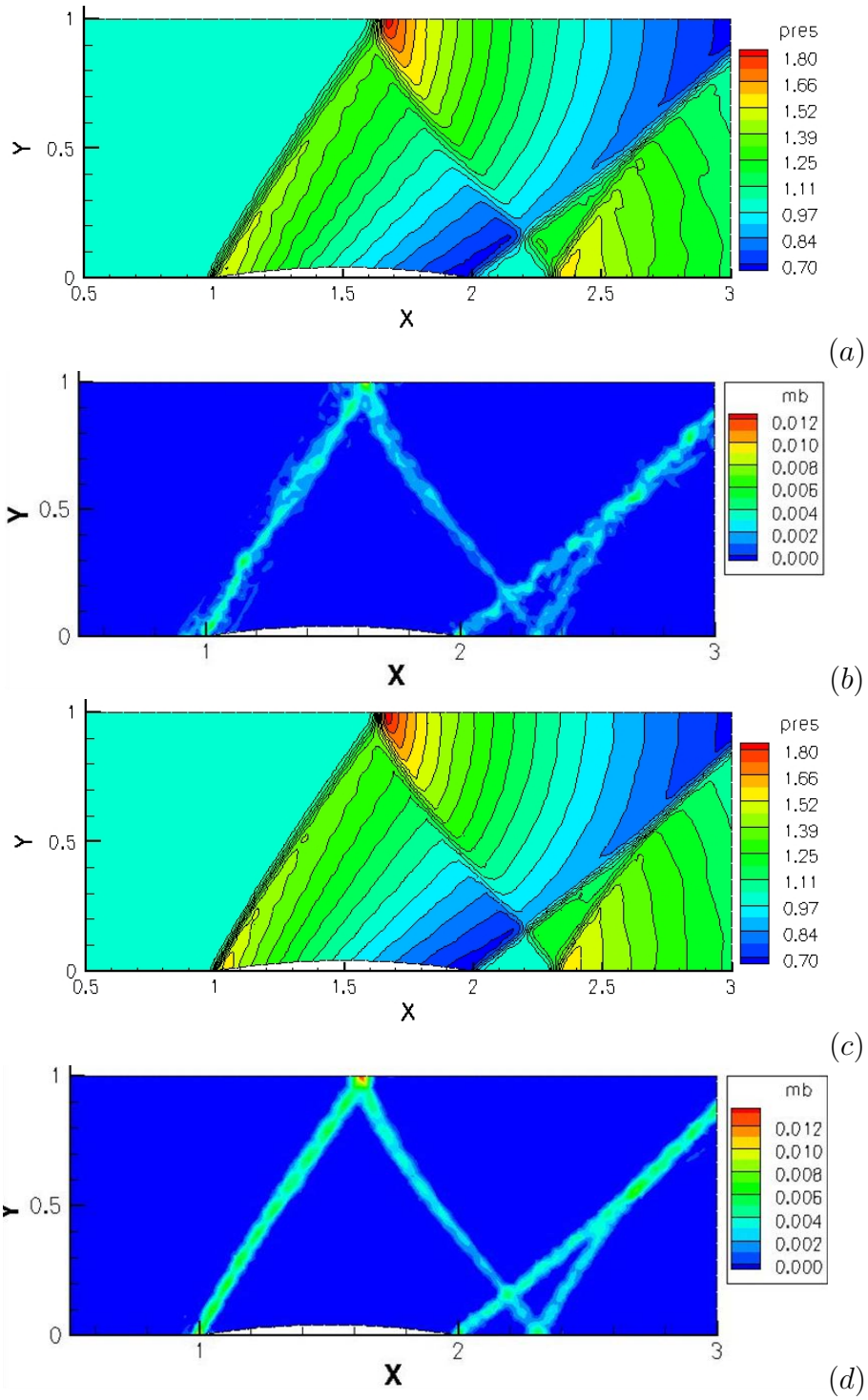
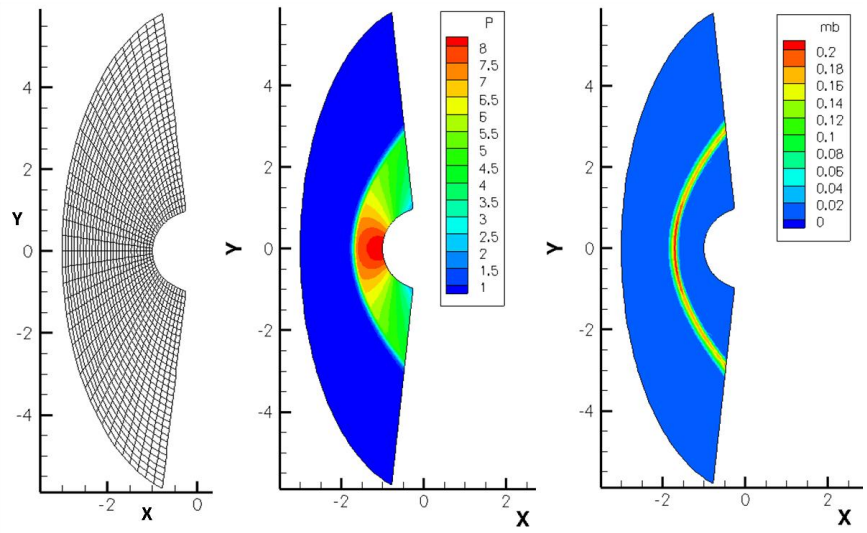
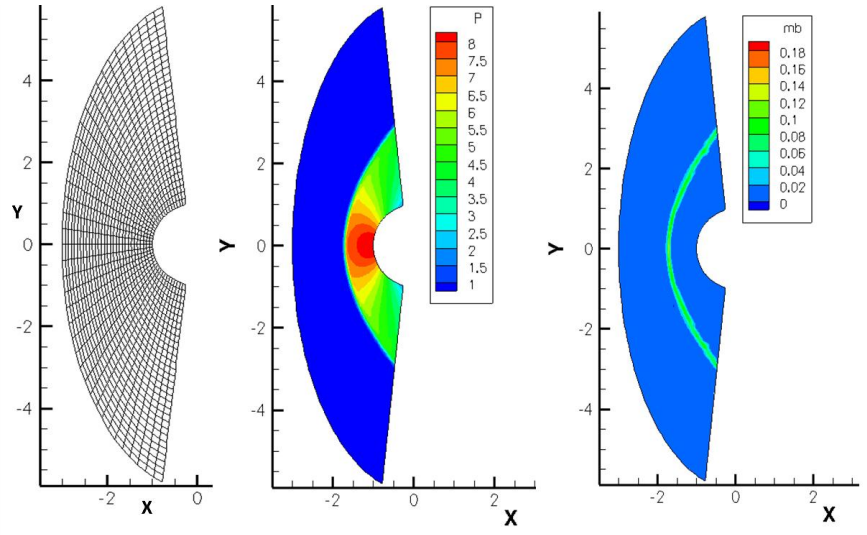


Figure 6.15: 3rd order SD computation with artificial viscosity on fully unstructured mesh (shown in figure 6.11). (a) Non dimensional pressure and (b) Artificial bulk viscosity contours using $r = 2$. (c) Non dimensional pressure and (d) Artificial bulk viscosity contours using $r = 0$.



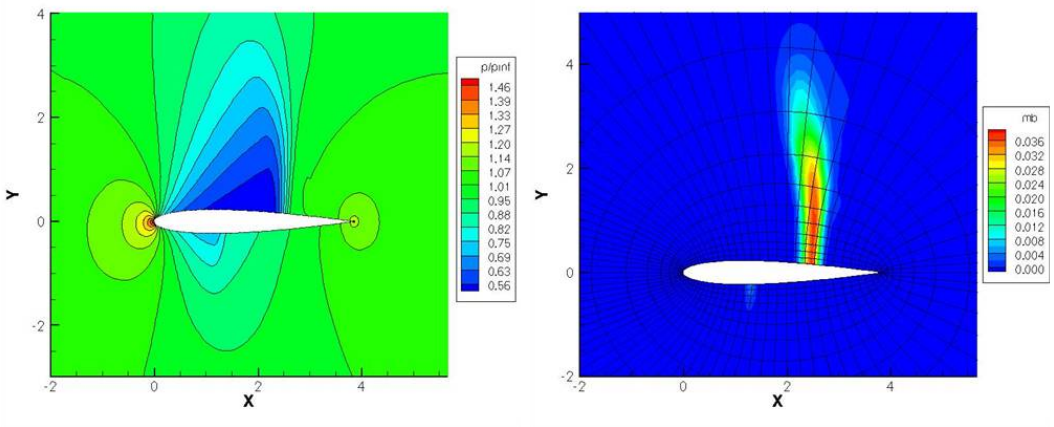
(a) (b) (c)

Figure 6.16: 3rd order computation of $M_\infty = 3.0$ flow past cylinder (a) 40x30 mesh (b) Non-dimensional pressure contours (c) Artificial bulk viscosity contours



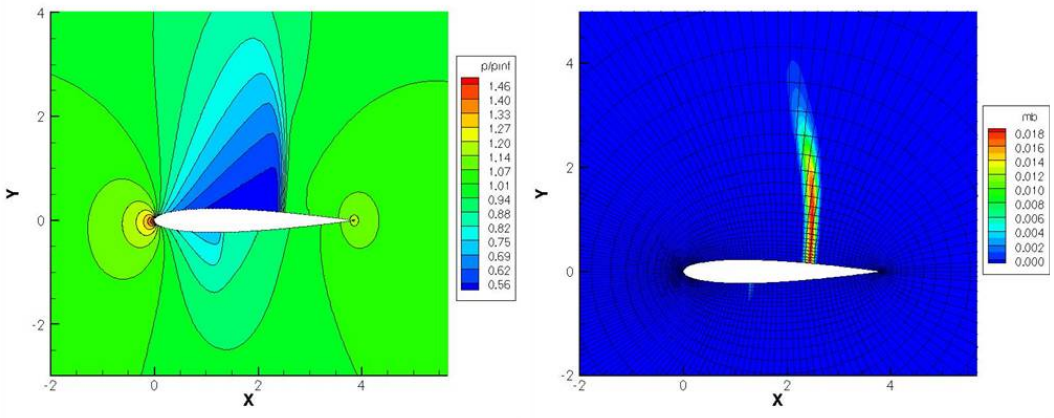
(a) (b) (c)

Figure 6.17: 4th order computation of $M_\infty = 3.0$ flow past cylinder (a) 40x30 mesh (b) Non-dimensional pressure contours (c) Artificial bulk viscosity contours



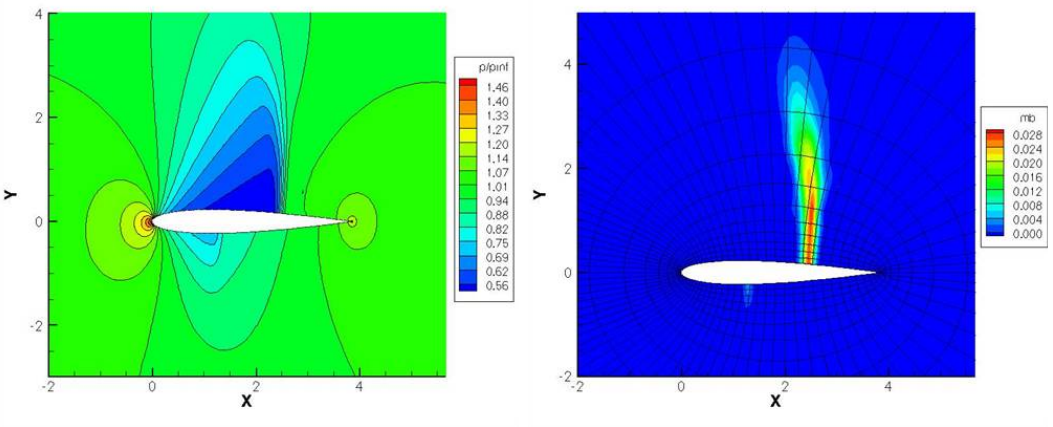
(a) (b)

Figure 6.18: Transonic flow past NACA0012 airfoil with M_∞=0.8 and $\alpha = 1.25$ degrees. 3rd order computation on coarse (80x16) mesh. (a) Pressure contours (b) Artificial bulk viscosity contours



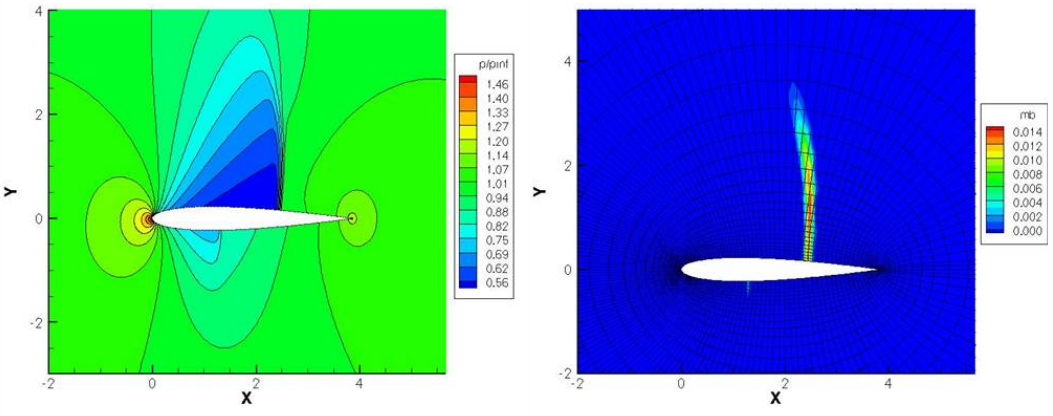
(a) (b)

Figure 6.19: Transonic flow past NACA0012 airfoil with M_∞=0.8 and $\alpha = 1.25$ degrees. 3rd order computation on fine (160x32) mesh. (a) Pressure contours (b) Artificial bulk viscosity contours



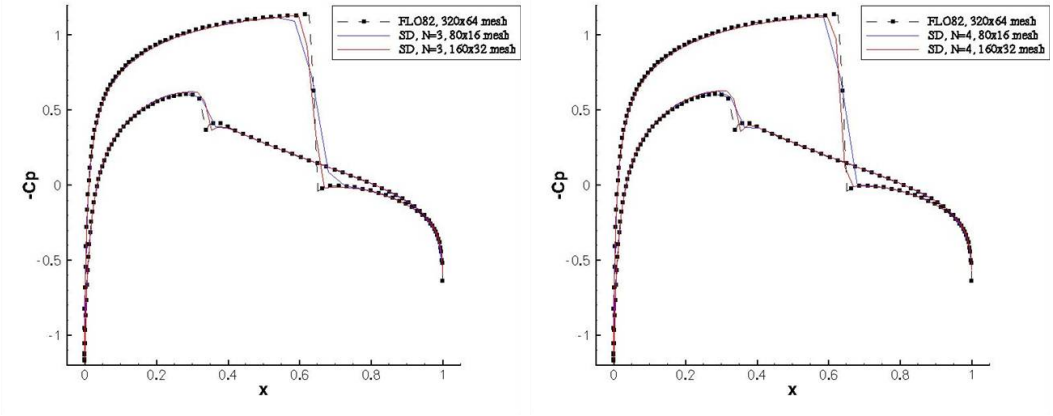
(a) (b)

Figure 6.20: Transonic flow past NACA0012 airfoil with M_∞=0.8 and $\alpha = 1.25$ degrees. 4th order computation on coarse (80x16) mesh. (a) Pressure contours (b) Artificial bulk viscosity contours



(a) (b)

Figure 6.21: Transonic flow past NACA0012 airfoil with M_∞=0.8 and $\alpha = 1.25$ degrees. 4th order computation on fine (160x32) mesh. (a) Pressure contours (b) Artificial bulk viscosity contours



(a) (b)

Figure 6.22: Transonic flow past NACA0012 airfoil with M_∞=0.8 and $\alpha = 1.25$ degrees. Comparison of Cp plots obtained using present method with those from FLO82 on a 320x64 mesh. (a) 3rd order computation (b) 4th order computation

Chapter 7

Local Mesh and Order Refinement

The present formulation suffers from the restriction that the calculation of a unique flux along a cell interface requires the flux points on either side of the interface to coincide. In other words, the flux polynomial approximation has to be conforming on either side of the interface. This has been valid for all the test-cases we have discussed so far. The limits imposed by the conforming restriction make it impossible to do local refinement by either subdividing existing cells (mesh refinement), or by increasing the polynomial order within selected cells (order refinement). To ease the conforming restriction, and allow for non-conforming configurations, we introduce the mortar element method [78, 16, 65], in which a one-dimensional polynomial function space called a mortar is defined along the sub-domain interfaces.

The basic idea of the mortar element method is that the mortar connects the neighboring sub-domains, and the sub-domains communicate only with the mortar and not with each other. In practice, a projection of the solution values is made from the contributing cell faces onto the mortar. It is on the mortar, and not on the sub-domain faces themselves, that the Riemann problem is solved to give a unique interface flux. The computed flux is then projected back onto the sub-domain faces. The projections from the sub-domain faces onto the mortar and back have to be carefully chosen, as two necessary conditions have to be satisfied. The first requirement is that the approximation retains global conservation. The second requirement is the outflow condition, which enforces the condition that the solution along the upwind

face be unchanged after projecting onto the mortar and back onto the face. Both conservation and outflow condition can be satisfied by a least squares matching of the face and mortar solution. The details of the least squares projection used are described in the following sections.

7.1 Mortar element method for order refinement

Local order refinement (p-refinement) results in non-conforming polynomial approximations on either side of the interface. Consider a sub-domain interface with left cell of order N^L and right cell of order N^R (see figure 7.1(a)). It is clear that the flux points on either side of the interface do not coincide with one another. To compute the interface fluxes, we use a method similar to that described by Kopriva [65]. We can define the solution approximations along the left and right sub-domain faces as

$$\begin{aligned} U^L(\xi) &= \sum_{j=1}^{N^L} U_j^L h_j^L(\xi) \in \mathbf{P}^{N^L-1} \\ U^R(\xi) &= \sum_{j=1}^{N^R} U_j^R h_j^R(\xi) \in \mathbf{P}^{N^R-1} \end{aligned} \tag{7.1}$$

where $\xi \in [0,1]$ is the local coordinate along the mortar element.

To be able to satisfy the outflow condition, the polynomial order of the mortar space must be sufficiently large to represent both $\mathbf{P}^{N^L-1}$ and $\mathbf{P}^{N^R-1}$. We choose the mortar polynomial order J=max(N^L, N^R). Now, the left and right solution approximations on the mortar can be represented by polynomials ϕ_L and ϕ_R that are defined as

$$\phi^{(L,R)}(\xi) = \sum_{j=1}^{J} \phi_j^{(L,R)} h_j^{\Xi}(\xi) \in \mathbf{P}^{J-1} \tag{7.2}$$

The computation of fluxes on the sub-domain faces using mortar elements involves three main steps -

1. Project the solution from sub-domain to mortar

2. Compute the fluxes on the mortar

3. Project the flux from mortar back to sub-domain

1. Sub-domain $\longrightarrow$ Mortar Projection

We have chosen the mortar order to be the larger of the polynomial orders of the two contributing sub-domains. For the sake on convenience, let us assume that $J = N^R$ which means that projection from right face to mortar is identity ($P^{R\rightarrow\Xi} = \mathbf{I}$, and $\phi^R = U^R$). For the least squares projection of the lower order space onto the mortar, we require that

$$\int_0^1 (\phi^L(\xi) - U^L(\xi)) h_m^{\Xi}(\xi) d\xi = 0, \qquad m = 1, 2, \cdots, J \tag{7.3}$$

Then the vector of the solution values along the mortar can be computed by

$$\Phi^L = P^{L\rightarrow\Xi} U^L = M^{-1} S U^L \tag{7.4}$$

where,

$$\begin{aligned} S_{mj} &= \int_0^1 h_m^{\Xi}(\xi) h_j^L(\xi) d\xi, && m = 1, \cdots, J, j = 1, \cdots, N^L \\ M_{mj} &= \int_0^1 h_m^{\Xi}(\xi) h_j^{\Xi}(\xi) d\xi, && m, j = 1, \cdots, J \end{aligned} \tag{7.5}$$

2. Flux calculation on the mortar elements

The fluxes on the mortar are computed as earlier because the solution polynomials on either side of the mortar are conforming. An approximate Reimann solver is used to compute the fluxes on the mortar. For viscous flow computations, the solution values are averaged on the mortar. We will denote the fluxes on the mortar by the variable Ψ.

3. Mortar $\longrightarrow$ Sub-domain projection

The computed mortar fluxes have to be projected back onto the sub-domain faces. Once again, for the right face, the projection matrix is identity and $F^R = \Psi^R$. To get the flux on the left face, the least squares projection requires that

$$\int_0^1 (F^L(\xi) - \Psi^L(\xi)) h_m^L(\xi) d\xi = 0, \qquad m = 1, 2, \cdots, N^L \tag{7.6}$$

The vector of the flux values on the left sub-domain face can be computed by

$$F^L = P^{\Xi \to \text{Ł}} \Psi^L = M^{-1} S U^L \tag{7.7}$$

where,

$$S_{mj} = \int_0^1 h_m^L(\xi) h_j^{\Xi}(\xi) d\xi, \qquad m = 1, \cdots, N^L, j = 1, \cdots, J \tag{7.8}$$

$$M_{mj} = \int_0^1 h_m^L(\xi) h_j^L(\xi) d\xi, \qquad m, j = 1, \cdots, N^L \tag{7.9}$$

7.2 Mortar element method for mesh refinement

Local mesh refinement (h-refinement) for quadrilateral meshes results in hanging nodes on cell faces (figure 7.1(b)). As a result, the flux points on the interface do not coincide with each other and the approximate Reimann solver cannot be applied directly. The treatment of such sub-domain refinement using mortar elements had been illustrated by Kopriva [65]. The current work uses a similar mortar method to deal with the non-conforming meshes obtained from local mesh refinement.

When a sub-domain is refined as shown in figure 7.1(b), two mortars are introduced corresponding to each of the short faces. This ensures that the outflow condition is satisfied. The outflow condition requires that the projection of face values from sub-domains Ω^2 and Ω^3 onto a mortar, and the subsequent projection back onto the faces returns the original polynomial functions.

We can define the solution approximations along the faces as

$$\begin{aligned} U^1(\xi) &= \sum_{j=1}^{N^1} U_j^1 h_j^1(\xi) \in \mathbf{P}^{N^1-1} \\ U^2(\xi) &= \sum_{j=1}^{N^2} U_j^2 h_j^2(\xi) \in \mathbf{P}^{N^2-1} \\ U^3(\xi) &= \sum_{j=1}^{N^3} U_j^3 h_j^3(\xi) \in \mathbf{P}^{N^3-1} \end{aligned} \tag{7.10}$$

where $\xi \in [0, 1]$ is the local sub-domain coordinate. We also define four mortar

functions

$$\begin{aligned}\phi^{1,(L,R)}(z) &= \sum_{j=1}^{J^1} \phi_j^{1,(L,R)} h_j^{\Xi^1}(z) \in \mathbf{P}^{J^1-1} \\ \phi^{2,(L,R)}(z) &= \sum_{j=1}^{J^2} \phi_j^{2,(L,R)} h_j^{\Xi^2}(z) \in \mathbf{P}^{J^2-1} \end{aligned} \tag{7.11}$$

which are functions of the local mortar coordinate $z \in [0,1]$. The superscripts L and R correspond to values on left and right of the mortar. We also define variables o^k and s^k to be the offset and the scale of the mortar with respect to the sub-domain Ω^k that contributes to it. Thus for $z \in [0,1]$, $\xi^k = o^k + s^k z$.

The orders of the mortar polynomials must be chosen sufficiently high so that the outflow condition is satisfied. This means the mortar must be at least as large as the largest sub-domain order of all contributing sub-domains. Thus, J^1=max(N^1, N^2) and J^2=max(N^1, N^3). It must be mentioned that for the present cases with mesh-refinement, the polynomial order in all the sub-domains are taken to be the same.

Like for the order refinement case, the computation of fluxes on the sub-domain faces using mortar elements involves three main steps.

1. Sub-domain $\longrightarrow$ Mortar Projections

To compute the sub-domain to mortar projection matrices, we use an un-weighted L^2 projection. Thus, we seek polynomials on the two mortars that best approximate the polynomial along the contributing face. For each mortar Ξ and each subdomain contributor Ω we require,

$$\int_0^1 (\phi(z) - U(\xi)) h_m^{\Xi}(z) dz = 0, \qquad m = 1, 2, \cdots, J \tag{7.12}$$

Then the vector of the solution values along the mortar can be computed by

$$\Phi = P^{\Omega \to \Xi} U = M^{-1} S U \tag{7.13}$$

where,

$$S_{mj} = \int_0^1 h_m^{\Xi}(z) h_j^{\Omega}(o^k + s^k z) dz, \qquad m = 1, \cdots, J, j = 1, \cdots, N \tag{7.14}$$

$$M_{mj} = \int_0^1 h_m^{\Xi}(z) h_j^{\Xi}(z) dz, \qquad m, j = 1, \cdots, J \tag{7.15}$$

2. Flux calculation on the mortar elements

The fluxes on the mortar are computed as earlier because the flux points on either side of the mortar coincide with each other. An approximate Reimann solver is used to compute the fluxes on the mortar. For viscous flow computations, the solution values are averaged on the mortar. We will denote the fluxes on the mortar by the variable Ψ.

3. Mortar $\longrightarrow$ Sub-domain projection

The computed fluxes on the mortar elements have to be projected back to the sub-domain faces. The current study involves computations where the polynomial order is same in all the cells. This means that the polynomial order on the mortar matches those on the short faces. Thus it is clear that the projection matrix for projection from mortar to the short faces is just the identity matrix, i.e, the computed fluxes are just copied from the mortar to the short faces. However, the projection from the mortars to the long face is a little more complicated. The piecewise polynomial that represents the fluxes along the mortars, possibly discontinuous, must be used to compute the continuous polynomial along the face. As before, we seek the best polynomial on the face that approximates the mortar solutions in the least squares sense. To obtain this least squares projection, we seek flux that satisfies.

$$\sum_{k=1}^{N^\Xi} \int_{o^k}^{o^k+s^k} (F^1(\xi) - \Psi^{\Xi^k}(\xi)) h_m^1(\xi) d\xi = 0 \tag{7.16}$$

The vector of values on the long sub-domain face can be computed by the projection

$$F^1 = \sum_{k=1}^{N^\Xi} P^k \Psi^{\Xi^k}, P^k = M^{-1} S^k \tag{7.17}$$

where the matrices M and S^k are defined as

$$\begin{aligned} M_{mj} &= \int_0^1 h_m^1(\xi) h_j^1(\xi) d\xi, & m, j = 1, \cdots, N^1 \\ S_{mj}^k &= s^k \int_0^1 h_m^1(o^k + s^k z) h_j^{\Xi^k}(z) dz, & m = 1, \cdots, N^1, j = 1, \cdots, J^k \end{aligned} \tag{7.18}$$

7.3 Results - Local Order refinement

The following test-cases intend to demonstrate that we can obtain a certain level of accuracy with lesser number of degrees of freedom using local/adaptive order refinement.

Inviscid flow past circle

The present test case involves steady subsonic inviscid flow past circle at $M_\infty = 0.2$. The 32×32 mesh used is shown in figure 7.2(a). The entropy error is taken as the indicator for solution accuracy. The entropy error at each solution point is defined as

$$\varepsilon_{sp} = \left(\frac{p}{p_\infty}\right)\left(\frac{\rho_\infty}{\rho}\right)^\gamma - 1 \tag{7.19}$$

Firstly, a 4^{th} order simulation was run, and the entropy error was computed at each solution point. The average entropy error (ε) within a cell was just taken as the average of the entropy errors (ε_{sp}) at the solution points within the cell. The maximum average entropy error (ε_{max}) was taken to be the largest value of average entropy error over the cells. For the 4^{th} order computation, this quantity was labeled $\varepsilon_{max}^{N=4}$, and was a measure of the accuracy of the computation.

The aim was to now obtain the same level of accuracy using different orders in different cells of the domain so as to reduce the required DOFs. A simple iterative algorithm was used to reduce the entropy errors until ε_{max} was lesser than $\varepsilon_{max}^{N=4}$. The steps are listed below.

Step 1. The computation is started with N=2 within all cells of the domain.

Step 2. Smoothing iterations are run until residual drops by 3 orders of magnitude.

Step 3. The average entropy error (ε) is computed for each cell.

Step 4. The maximum average entropy error $\varepsilon_{max}^{Nvarying}$ is obtained for the domain.

Step 5. If $\varepsilon_{max}^{Nvarying} \leq \varepsilon_{max}^{N=4}$, the residual is dropped to 10^{-10} and the simulation is terminated. Otherwise go to step 6.

Step 6. If for a cell, $\varepsilon > 0.9\varepsilon_{max}^{N=4}$, the order within the cell is increased by 1.

Step 7. If for a cell, $\varepsilon < 0.1\varepsilon_{max}^{N=4}$, the order within the cell is decreased by 1.

Step 8. Go to step 2.

Starting with N=2 within all cells in the domain, figure 7.2(b), (c) and (d) shows the variation of N after one, two and three levels of order refinement/coarsening. After the third level of order refinement/coarsening, the accuracy criterion (in step 5) was satisfied, and simulation is run till the residual drops to 10^{-10}. Figure 7.3(a) compares the residual drop for the N=4 computation and the computation with varying order. The computation with adaptive order refinement is about twice as fast as the computation with constant N, to obtain the same level of accuracy. Also the number of DOFs required to obtain a given accuracy level is much lesser with adaptive order variation (7512 DOFs vs. 16384 DOFs for 4th order simulation). The Mach contours obtained are shown in Figure 7.3(b).

This simple test-case demonstrates the advantage of adaptive order refinement in the context of steady flow problems. It also clearly indicates that to obtain a specified accuracy level, it is not necessary to use high order in all regions of the domain. Thus, we can significantly save in computational effort and time by using adaptive order refinement.

Viscous flow past circle

This test case involves flow at $M_\infty = 0.2$ and $Re = 100$ past a circle (identical to test-case in Section 4.2). In the present computation, different polynomial orders are used in different regions of the computational domain (as shown in figure 7.4)(a)). Fourth order cells are used in the region close to the cylinder and 2nd order cells are used near the free stream boundaries away from the cylinder.

The force coefficients obtained are compared to a simulation with fourth order cells in the entire domain. It is the clear that the force coefficient plot obtained is almost identical for both cases (see figure 7.4)(c)). Also the number of degrees of freedom when using varying order is significantly reduced (14596 DOFs vs. 21376 DOFs for fourth order in entire domain). This also reduces the computational time per iteration by about 40%. Thus, for cases where we are interested in the lift-drag characteristics of a body, we can obtain desired accuracy at reduced computational cost by varying the polynomial order.

7.4 Results - Local Mesh Refinement

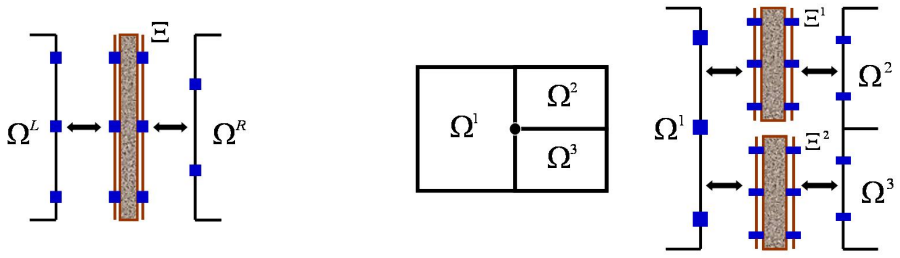
The mortar element method described above has been used to enable adaptive mesh-refinement for computations with shocks. The criterion for mesh refinement used here is non-zero artificial viscosity. This is a simple and sufficient criterion since we are looking to refine the mesh only in the region of shocks.

Supersonic flow past cylinder

In this test-case, adaptive mesh refinement is applied to the $M_\infty = 3.0$ flow past cylinder. The details of this case have been discussed in Section 6.2. The present computation uses 3^{rd} order SD. Two levels of mesh-refinement are used. Figure 7.5 shows the mesh after successive levels of mesh refinement. The pressure contours and AV contours obtained after local mesh-refinement are shown in figure 7.6 and figure 7.7 respectively. It is observed that as the mesh is refined, the computed shock profiles become thinner, the amount of artificial viscosity added is lesser, and the pressure contours become more accurate. Figure 7.8 shows the center-line pressure variation and compares the present results with those obtained by Kawai and Lele [63] using 6th order Compact Difference with artificial viscosity (on an 80×60 mesh). The pressure profiles using 3rd order SD on a 40×30 mesh compare well with those obtained by Kawai and Lele. A much sharper shock jump profile is obtained after two levels of mesh refinement. The computed shock location agrees very well with the predicted value of x=-1.7.

Supersonic flow past bump

Adaptive mesh refinement is also applied to the supersonic bump case described earlier in Section 6.2. A 4^{th} order SD calculation is used. Once again the criterion for mesh refinement is non-zero AV. Figures 7.9 (a) and (b) show the computational mesh after one and two levels of refinement respectively. Figures 7.10(a)-(c) show the pressure contours obtained for successive levels of mesh refinement. Figures 7.11(a)-(c) show the artificial viscosity contours obtained for the corresponding cases.



(a) (b)

Figure 7.1: (a) Mortar element for local order refinement (b) Hanging node generated from local mesh refinement and corresponding mortar elements for each child face on interface

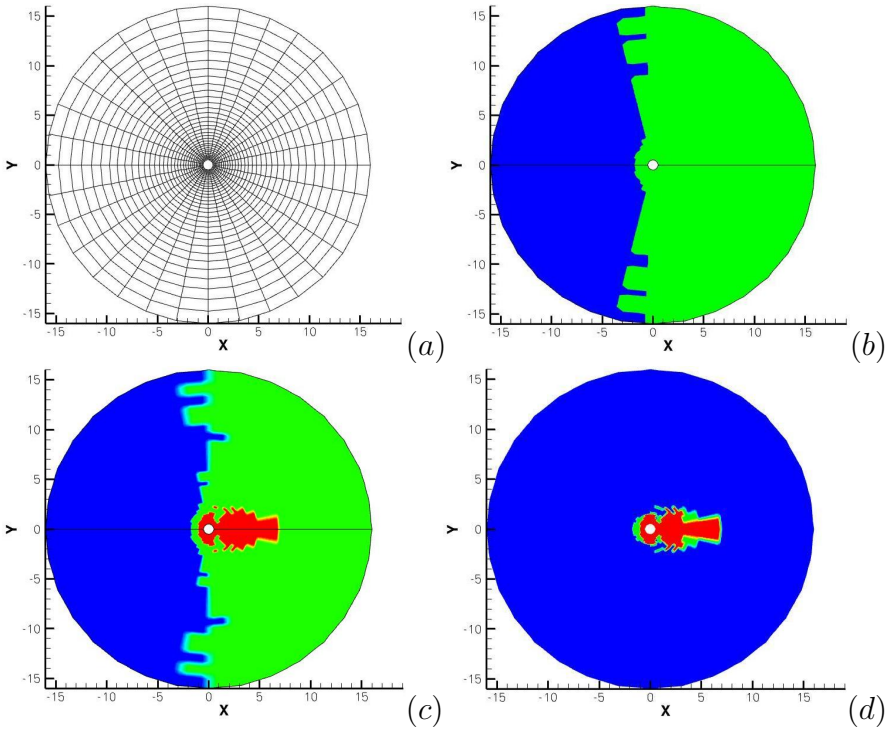
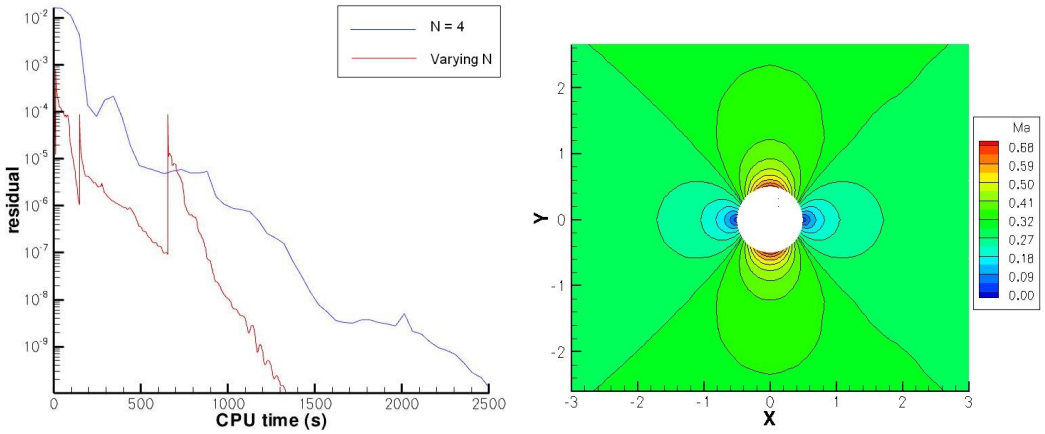
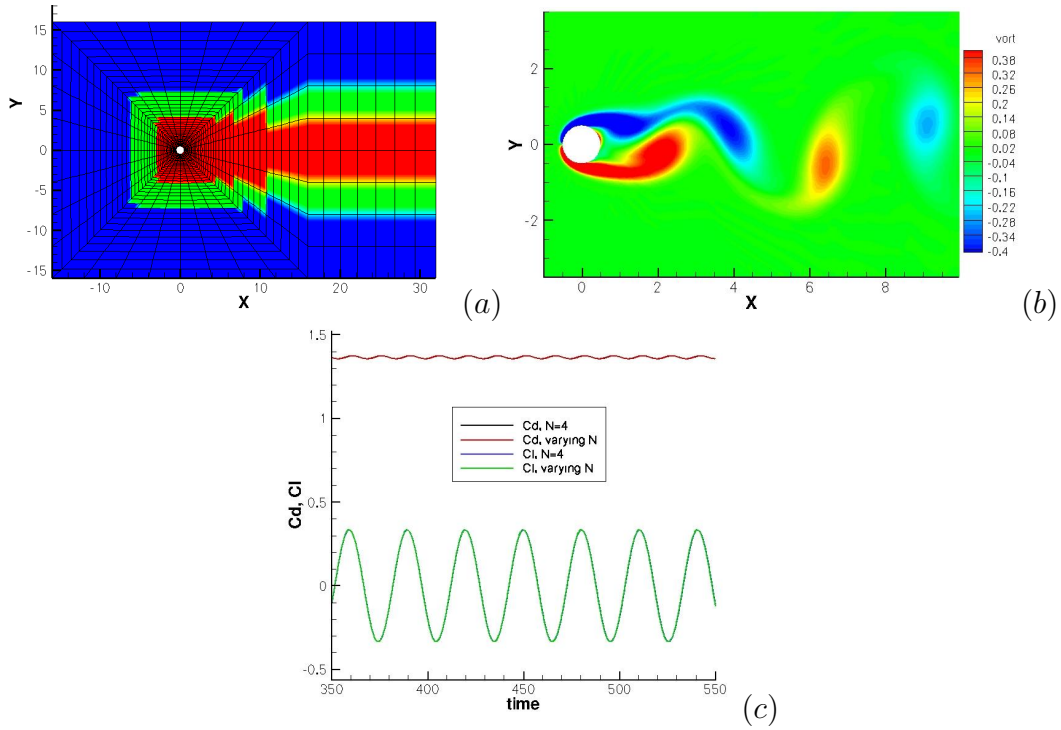


Figure 7.2: Variation of polynomial order for inviscid flow past a circle test case (a) 32×32 mesh (b) After 1st refinement step (c) After 2nd refinement step (d) After 3rd (last) refinement step. *Red* $\Leftrightarrow 4^{th}$ order cells, *Green* $\Leftrightarrow 3^{rd}$ order cells, *Blue* $\Leftrightarrow 2^{nd}$ order cells



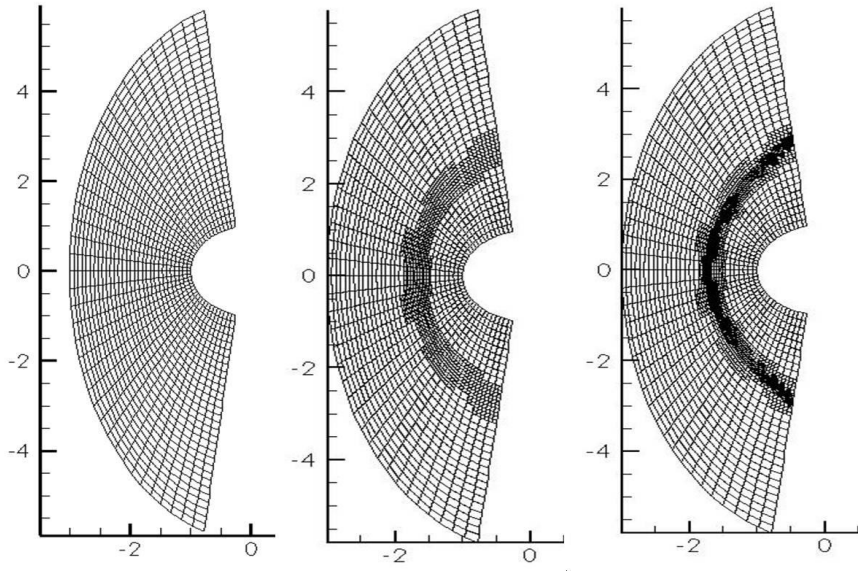
(a) (b)

Figure 7.3: (a) Comparison of convergence with and without local order refinement (b) Mach contours obtained with order refinement



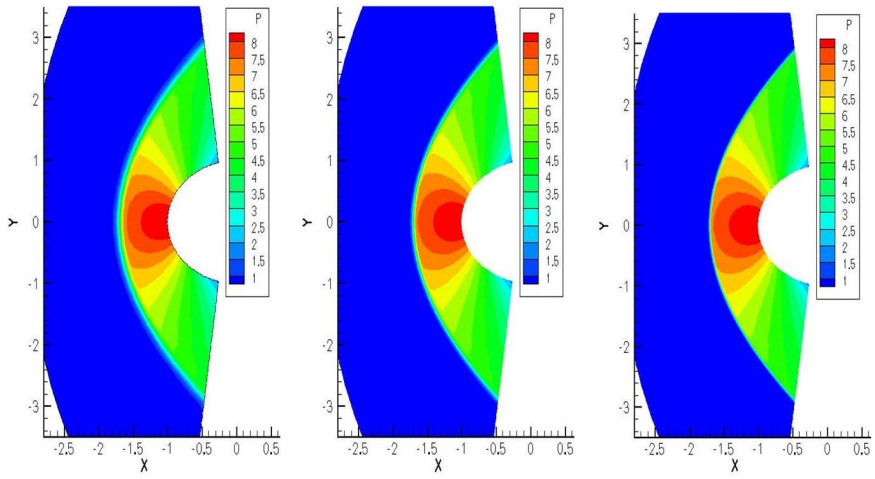
(a) (b) (c)

Figure 7.4: Re=100 flow past circle (a)Variation of polynomial order (N) within the domain *Red* $\Leftrightarrow 4^{th}$ order cells, *Green* $\Leftrightarrow 3^{rd}$ order cells, *Blue* $\Leftrightarrow 2^{nd}$ order cells, (b) Vorticity contours obtained (c) Lift and Drag coefficients



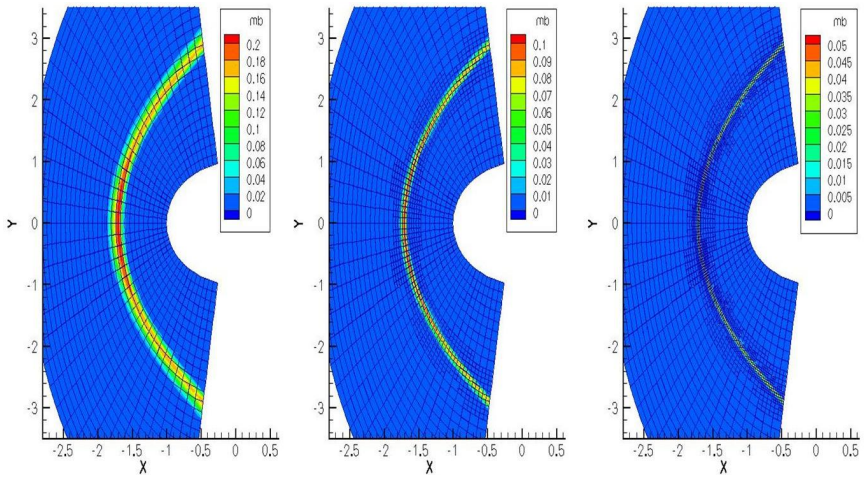
(a) (b) (c)

Figure 7.5: Successive levels of mesh refinement for $M_\infty = 3.0$ flow past cylinder test-case (a) 40x30 initial mesh (b) One level mesh refinement (c) Two levels of mesh refinement



(a) (b) (c)

Figure 7.6: Pressure contours for 3rd order computation of $M_\infty = 3.0$ flow past cylinder (a) 40x30 initial mesh (b) One level mesh refinement (c) Two levels of mesh refinement



(a) (b) (c)

Figure 7.7: Artificial bulk viscosity contours for 3rd order computation of $M_\infty = 3.0$ flow past cylinder (a) 40x30 initial mesh (b) One level mesh refinement (c) Two levels of mesh refinement

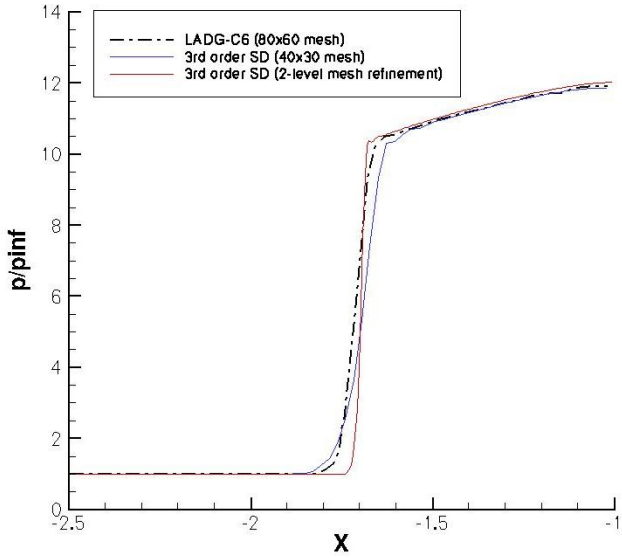


Figure 7.8: Comparison of centerline pressure obtained using 3rd order SD with that obtained using 6th order Compact Difference and artificial viscosity on similar mesh [63]

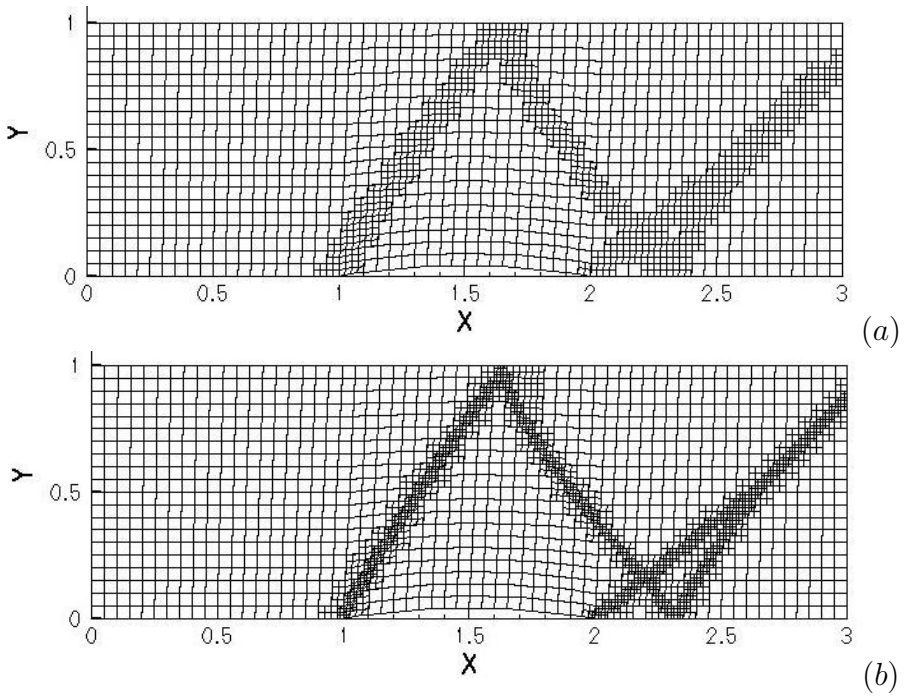


Figure 7.9: Computational grid for supersonic flow past bump after (a) one level of mesh refinement (b) two levels of mesh refinement

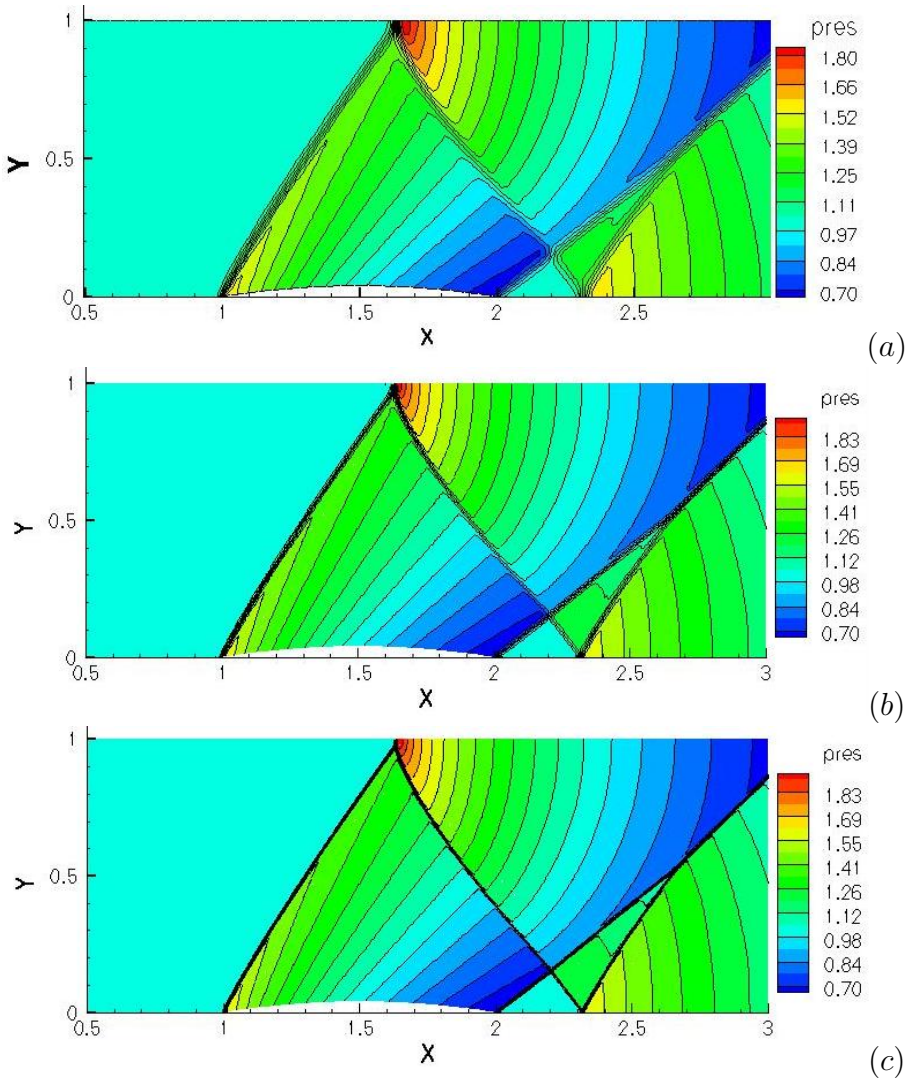


Figure 7.10: Non-dimensional pressure contours for 4th order computation (a) on initial 60×20 mesh (b) after 1 level mesh refinement (c) after 2 levels mesh refinement

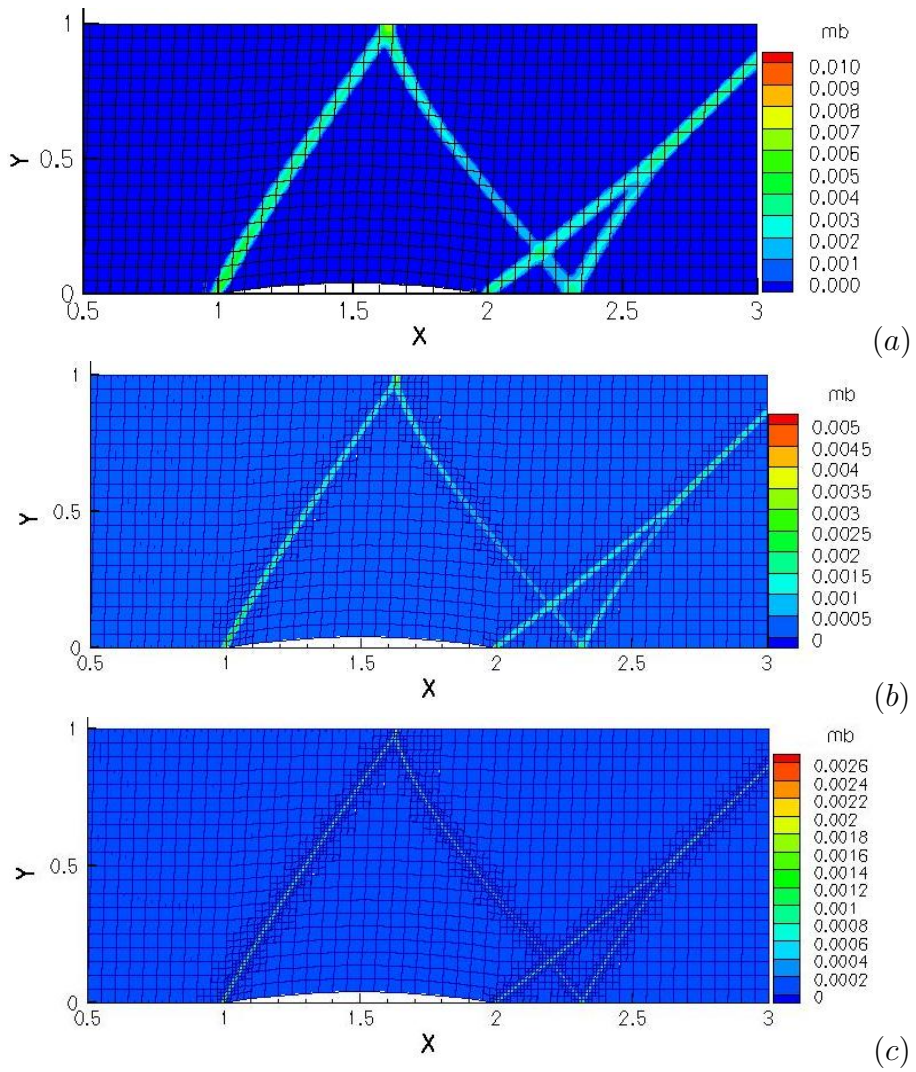


Figure 7.11: Artificial viscosity contours for 4th order computation (a) on initial 60×20 mesh (b) after 1 level mesh refinement (c) after 2 levels mesh refinement

Chapter 8

Conclusions and Future work

8.1 Summary and Conclusions

Spectral Difference Method

The present thesis work makes an effort towards the realization of a high-order accurate flow solver for unstructured mesh computations, that is also efficient and robust. As part of this work, a 2D flow solver was developed for viscous compressible flows using unstructured quadrilateral meshes. The high order spatial discretization algorithm used was the Spectral Difference method. The solver was tested and validated against a variety of steady and unsteady, inviscid and viscous flow problems with existing experimental or numerical data. Test cases with analytical solutions were used to demonstrate the order of accuracy of the spectral difference method. All the numerical computations have been conducted on 2D test cases, but the present numerical scheme, as well as the relevant computational tools that have been developed and implemented as part of this work, can be easily extended to 3D. The solver has been successfully applied to the study of problems with vortex dominated flows by Liang et al. [72, 71], and extended to the study of flows with moving and deforming meshes by Kui et al. [90]

The Spectral Difference scheme for quadrilateral (and hexahedral) cells is very promising, since they are relatively cheap and easy to implement for general solution as well as geometrical polynomial orders. They are capable of achieving a very good

accuracy on relatively coarse meshes and are more suitable than triangular meshes for boundary layer flows. The absence of a firm mathematical basis and theoretical proof of stability for the SD scheme (until recently), has led to some speculation about the stability of the SD method. However, the present work shows that the SD method on quads is stable for a wide range of problems for both inviscid and viscous flows. Different approaches for the discretization of the viscous terms have been used in the context of DG method [9, 11]. However, in the present work, the relatively simpler average approach [66] is used with satisfactory results.

Convergence acceleration

The issue of the slow convergence of high-order schemes with explicit time-stepping schemes has been addressed. In order to obtain convergence acceleration, both the p-multigrid method and implicit time-stepping has been implemented. The speed up obtained has been analyzed for steady and unsteady, viscous and inviscid flow problems. Earlier efforts using p-multigrid with Spectral Difference were restricted to up to 3rd order accurate computations on triangular meshes. The present work applies the p-multigrid method for up to 4th order accurate computations. The p-multigrid method and Backward Euler Implicit LU-SGS were used to accelerate convergence for problems with steady state solution. To handle unsteady flow problems, an Implicit BDF2 LU-SGS algorithm was used, which is second order accurate in time. The time-accurate results obtained compare well with those obtained using explicit RK schemes.

The p-multigrid method was found to give reasonable speed-up over explicit time-stepping. It was relatively simple to implement and can be easily extended to 3D formulations. However the p-multigrid method suffered from certain drawbacks. The convergence of the p-multigrid algorithm with explicit smoothers was dependent on optimizing the time-step sizes at each level. Also, it was not a trivial exercise to determine these parameters for optimal convergence, as they were generally dependent on the problem as well as the solution polynomial order. Also the p-multigrid algorithm was applicable only to problems with steady state solutions.

The speed-up obtained using implicit time-stepping was significantly higher as

compared to p-multigrid for all the test-cases considered. Obtaining speed-up using implicit schemes was made much easier (compared to p-multigrid method) because one set of parameters seemed to work quite effectively for a variety of problems. Using the LU-SGS algorithm ensured that the memory requirements were relatively small. Furthermore, implementation of the Implicit BDF2 scheme allowed us to reduce computational times for unsteady flow problems. However, achieving an efficient implementation of the Implicit LU-SGS algorithm was far from trivial. Particular care and effort was required to efficiently implement the steps involving computation of the element gradient matrix and updating solutions during forward-backward Gauss-Seidel sweeps. Overall, implicit time-stepping proved to be more effective and promising than the p-multigrid method for accelerating convergence of high-order formulations.

Shock capturing using artificial viscosity

The present work combines artificial viscosity with the Spectral difference formulation to enable computation of flows with shocks. The shock-capturing properties of the proposed formulation are demonstrated with test cases in 1D and 2D. Promising results have been obtained for these cases, with the method being able to produce a stable solution with sharp resolution of shocks, and no significant spurious oscillations. An element-based restriction-prolongation filter has been developed to ensure smooth variation of artificial viscosity.

Initial efforts were made to extend the high wave-number biased artificial viscosity approach to an unstructured grid setup. But results show that computation of the higher order derivatives of the sensor quantities results in the introduction of numerical errors which affects the shock capturing properties adversely. Alternately, the proposed form of AV scales the bulk viscosity coefficient directly as the dilatation sensor. In combination with a smoothing filter, and a switch to turn off AV in smooth regions of the flow, this method is capable of producing stable, sharply resolved shocks without any significant spurious oscillations. The AV coefficient is $O(\Delta)$ in the region of the discontinuity and the shock is found to be captured over one to two cells. Earlier efforts by other researchers on shock capturing using the SD schemes with limiters, suffered from issues such as the inability to converge to

machine zero, due to limit cycle oscillations. However with the present AV scheme, all the test cases presented converged to machine zero.

Local mesh and order refinement

Adaptive mesh and order refinement have great potential in reducing the computational effort required to reach a specified level of accuracy, particularly in the context of high-order formulations. A mortar element method has been implemented to deal with non-conforming solution approximations at cell interface, that arise from order and/or mesh refinement in quadrilaterals. The benefits of varying the polynomial order have been demonstrated for steady and unsteady flow problems. Local h-refinement has been used in combination with artificial viscosity to enhance the shock capturing capabilities of the present solver. The capability to handle hanging nodes generated from mesh refinement in quadrilaterals also adds great flexibility to the scheme, with regards to mesh generation and accuracy improvement.

The present work demonstrates using simple test cases that the use of varying polynomial order within different regions of the domain can help attain a certain level of accuracy using lesser degrees of freedom. It would be safe to conclude that p-refinement can significantly reduce computational effort and time for high order flow computations on complex geometries, especially in 3D. The present work also shows that adaptive h-refinement is an efficient and effective way to improve accuracy for the computation of flows with shocks.

8.2 Future Work

SD on simplex meshes

Earlier efforts by other researchers with the Spectral Difference Method on triangular meshes have shown stability issues for 4th and higher order accurate formulations. This is an important issue that remains to be resolved for the usability of the SD method for real-life problems on complex geometries. However recent ongoing work has investigated the use of Raviart-Thomas polynomials as basis functions for triangles, with promising results. Achieving stable discretizations for the Spectral

Difference formulation on triangular and tetrahedral meshes can add great flexibility to solution of complex flow problems, particularly with regards to mesh generation, as well as the ability to use hybrid meshes.

Extension to 3D flow problems

The advantage of using high order methods over lower order solvers becomes evident for flow problems requiring high accuracy, such as turbulent flow, flows with complex vortical interactions, turbulent combustion and aero-acoustics. Such flow problems are essentially three-dimensional in nature. It is therefore necessary to develop an efficient and robust 3D flow solver that can be applied to complex flow geometries. The present SD formulation has been extended to hexahedral meshes with promising results. The supporting framework developed as part of the present thesis - including convergence acceleration techniques, shock capturing, and adaptive *hp*-refinement can be easily extended to 3D, and will be a focus of future efforts.

Further application testing of shock capturing approach

In this work, the proposed artificial viscosity model has demonstrated good shock capturing properties for 3rd and 4th order SD computations. However, the test cases contained in this thesis were limited to a few applications. Also, the test cases considered were inviscid flow problems with shocks. Within the context of the compressible Navier-Stokes equations, the present AV approach should be explored for unsteady flow problems, turbulent flows, and a greater selection of more complex transonic, supersonic and hypersonic flow cases.

Dynamic mesh and order adaptation

The solver can be extended to dynamically increase/decrease order as well as refine/coarsen mesh during the course of the simulation. For this purpose it is necessary to modify the solver to enable the creation/deletion of cells, faces and nodes dynamically. Furthermore, adding an hp-adaptive capability should allow for better degree of freedom management and optimization in the adaptation process. In smooth flow regions, refinement could be done with p and near discontinuities refinement could

be done with h. Different strategies for adaptation such as gradient based, error based and output based adaptation should be investigated to determine an optimal approach for adaptive *hp*-refinement.

Bibliography

[1] R. Abgrall. On essentially non-oscillatory schemes on unstructured meshes: analysis and implementation. *Journal of Computational Physics*, 114:45–58, 1994.

[2] S. R. Allmaras. *A coupled Euler/Navier-Stokes algorithm for 2-D unsteady transonic shock/boundary-layer interaction*. PhD thesis, Massachusetts Institute of Technology, 1989.

[3] S. R. Allmaras and M. B. Giles. A second-order flux split scheme for the unsteady 2-d euler equations on arbitrary meshes. 1987. AIAA Paper 87-1119.

[4] H. L. Atkins and C. W. Shu. Quadrature-free implementation of discontinuous galerkin method for hyperbolic equations. *AIAA Journal*, 36:775–782, 1998.

[5] I. Babuska, B. A. Szabo, and I. N. Katz. The p-version of the finite element method. *SIAM Journal of Numerical Analysis*, 18:515–545, 1981.

[6] B. S. Baldwin and R. W. MacCormack. Interaction of strong-shock wave with turbulent boundary layer. 1974. AIAA Paper 74-558.

[7] G.E. Barter and D.L. Darmofal. Shock capturing with higher-order, pde-based artificial viscosity. 2007. AIAA Paper 2007-3823.

[8] T. J. Barth. Recent developments in high-order k-exact reconstruction on unstructured meshes. 1993. AIAA Paper 93-0668.

[9] F. Bassi and S. Rebay. A high-order discontinuous finite element method for the numerical solution of the compressible navier-stokes equations. *Journal of Computational Physics*, 131:267–279, 1997.

[10] F. Bassi and S. Rebay. *Discontinuous Galerkin Methods: Theory, Computation and Applications*, pages 197–208. Springer, 2000.

[11] F. Bassi and S. Rebay. Numerical evaluation of two discontinuous galerkin methods for the compressible navier-stokes equations. *International Journal for Numerical Methods in Fluids*, 40:197–207, 2002.

[12] F. Bassi and S. Rebay. Numerical solution of the euler equations with a multi-order discontinuous finite element method. 2002. Second International Conference on Computational Fluid Dynamics, Sydney, Australia.

[13] F. Bassia, A. Crivellini, S. Rebay, and M. Savini. Discontinuous galerkin solution of the reynolds-averaged navier-stokes and k2o turbulence model equations. *Journal of Computational Fluids*, 34:507–540, 2005.

[14] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. In *Acta Numerica, edited by A. Iserles.*

[15] M. J. Berger and A. Jameson. Automatic adaptive grid refinement for the euler equations. *AIAA Journal*, 23:561–568, 1985.

[16] C. Bernardi, Y. Maday, and A. T. Patera. A new nonconforming approach to domain decomposition: The mortar element method. In *Nonlinear Partial Differential Equations and their Applications, edited by H. Brezis and J. L. Lions.*

[17] A. Bhagatwala and S.K. Lele. A modified artificial viscosity approach for compressible turbulence simulations. *Journal of Computational Physics*, 228:4965–4969, 2009.

[18] H. M. Blackburn and S. Schmidt. Spectral element filtering tecniques for large eddy simulation with dynamic estimation. *Journal of Computational Physics*, 186:610–629, 2003.

[19] F. Brezzi, G. Manzini, D. Marini, P. Pietra, and A. Russo. Discontinuous galerkin approximations for elliptic problems. *Numerical Methods for Partial Differential Equations*, 16:365–378, 2000.

[20] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods in Fluid Dynamics*. Springer-Verlag, 1987.

[21] R. F. Chen and Z. J. Wang. Fast block lower-upper symmetric gauss seidel scheme for arbitrary grids. *AIAA Journal*, 38:2238–2245, 2000.

[22] B. Cockburn and C. W. Shu. Tvb runge-kutta local projection discontinuous galerkin finite element method for scalar conservation laws ii: General framework. *Journal of Mathematical Computation*, 52:411–435, 1989.

[23] B. Cockburn and C. W. Shu. The runge-kutta local projection p1-discontinuous galerkin method for scalar conservation laws. *RAIRO Model Math. Anal. Numer.*, 25:337–361, 1991.

[24] B. Cockburn and C. W. Shu. The local discontinuous galerkin method for time-dependent convection-diffusion systems. *SIAM Journal of Numerical Analysis*, 35:2440–2463, 1998.

[25] B. Cockburn and C. W. Shu. Runge-kutta discontinuous galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, 16:173–261, 2001.

[26] A. W. Cook and W. H. Cabot. A high-wavenumber viscosity for high-resolution numerical methods. *Journal of Computational Physics*, 195:594–601, 2004.

[27] A. W. Cook and W. H. Cabot. Hyperviscosity for shock-turbulence interactions. *Journal of Computational Physics*, 203:379–385, 2005.

[28] H. Ding, C. Shu, K. S. Yeo, and D. Xu. Numerical simulation of flows around two circular cylinders by mesh-free least square-based finite difference methods. *International Journal for Numerical Methods in Fluids*, 53:305–332, 2007.

[29] L. J. Durlofsky, B. Enquist, and S. Osher. Triangle based adaptive stencils for the solution of hyperbolic conservation laws. *Journal of Computational Physics*, 98:64–88, 1992.

[30] G. Erlebacher, M. Y. Hussaini, and C. W. Shu. Interaction of a shock with a longitudinal vortex. *Journal of Fluid Mechanics*, 337:129–153, 1997.

[31] K. J. Fidkowski and D. L. Darmofal. A triangular cut-cell adaptive method for high-order discretizations of the compressible navier-stokes equations. *Journal of Computational Physics*, 225:1653–1672, 2007.

[32] K. J. Fidkowski, T. A. Oliver, J. Lu, and D. L. Darmofal. p-multigrid solution of high-order discontinuous galerkin discretizations of the compressible navier-stokes equations. *Journal of Computational Physics*, 207:92–113, 2005.

[33] K.J. Fidkowski and P.L. Roe. Entropy-based mesh refinement, i: The entropy adjoint approach. 2009. AIAA Paper 2009-3790.

[34] B. Fiorina and S.K. Lele. An artificial nonlinear diffusivity method for supersonic reacting flows with shocks. *Journal of Computational Physics*, 222:246–264, 2007.

[35] O. Friedrich. Weighted essentially non-oscillatory schemes for the interpolation of mean values on unstructured grids. *Journal of Computational Physics*, 144:194–212, 1998.

[36] M. Giles and N. Pierce. Adjoint error correction for integral outputs. In *Lecture Notes in Computational Science and Engineering: Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics, Vol 25.*

[37] D. Gottlieb and S. A. Orszag. *Numerical Analysis of Spectral Methods.* SIAM, 1977.

[38] T. Haga, K. Sawada, and Z. J.Wang. An implicit lu-sgs scheme for spectral volume method on unstructured tetrahedral grids. *Communications in Computational Physics*, 6:978–996, 2009.

[39] R. Harris and Z. J. Wang. High-order adaptive quadrature-free spectral volume method on unstructured grids. 2008. AIAA Paper 2008-779.

[40] R. Harris, Z. J. Wang, and Y. Liu. Efficient quadrature-free high order spectral volume method on unstructured grids: Theory and 2d implementation. *Journal of Computational Physics*, 227:1620–1642, 2008.

[41] A. Harten and S. Chakravarthy. Multi-dimensional eno schemes for general geometries. 1991. ICASE Report No. 91-76.

[42] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy. Uniformly high-order essentially non-oscillatory schemes iii. *Journal of Computational Physics*, 71:231–256, 1987.

[43] R. Hartmann and P. Houston. Adaptive discontinuous galerkin finite element methods for the compressible euler equations. *Journal of Computational Physics*, 183:508–532, 2002.

[44] B. T. Helenbrook and H. L. Atkins. Application of p-multigrid to discontinuous galerkin formulations of the poisson equation. *AIAA Journal*, 44:566–575, 2005.

[45] B. T. Helenbrook, D. J. Mavriplis, and H. L. Atkins. Analysis of pmultigrid for continuous and discontinuous finite element discretizations. 2003. AIAA Paper 2003-3989.

[46] D. G. Holmes, S. Lamson, and S. D. Connell. Quasi-3d solutions for transonic, inviscid flours by adaptive triangulation. 1988. ASME Paper 88-GT-83.

[47] D. G. Holmes and S. H. Lamson. *Adaptive Triangular Meshes for Compressible Flow Solutions.*

[48] C. Hu and C. W. Shu. Weighted essentially non-oscillatory schemes on triangular meshes. *Journal of Computational Physics*, 150:97–127, 1999.

[49] H. T. Huynh. A flux reconstruction approach to high-order schemes including discontinuous galerkin methods. 2007. AIAA Paper 2007-4079.

[50] H. T. Huynh. A reconstruction approach to high-order schemes including discontinuous galerkin for diffusion. 2009. AIAA Paper 2009-0403.

[51] A. Jameson. Transonic potential flow calculations using conservation form. In *Proceedings of Second AIAA Computational Fluid Dynamics Conference, Hartford*.

[52] A. Jameson. Solution of the euler equations by a multigrid method. *Applied Mathematics and Computation*, 13:327–356, 1983.

[53] A. Jameson. Analysis and design of numerical schemes for gas dynamics, 1: Artificial diffusion, upwind biasing, limiters and their effect on accuracy and multigrid convergence. *International Journal of Computational Fluid Dynamics*, 4:171–218, 1994.

[54] A. Jameson. Analysis and design of numerical schemes for gas dynamics, 2: Artificial diffusion and discrete shock structure. *International Journal of Computational Fluid Dynamics*, 5:1–38, 1995.

[55] A. Jameson. A proof of the stability of the spectral difference method for all orders of accuracy. *Journal of Scientific Computing*, 2009. DOI:10.1007/s10915-009-9339-4.

[56] A. Jameson, W. Schmidt, and E. Turkel. Numerical simulation of the euler equations by finite volume methods using runge-kutta time stepping schemes. 1981. AIAA Paper 81-125.

[57] A. Jameson and S. Yoon. Lowerupper implicit schemes with multiple grids for the euler equations. *AIAA Journal*, 25:929–935, 1987.

[58] G. Jiang and C. W. Shu. Efficient implementation of weighted eno schemes. *Journal of Computational Physics*, 126:202–220, 1996.

[59] K. D. Jones, C. M. Dohring, and M. F. Platzer. Experimental and computational investigation of the knoller-betz effect. *AIAA Journal*, 36:780–783, 1998.

[60] Z. J.Wang, Y. Sun, C. Liang, and Y. Liu. Extension of the sd method to viscous flow on unstructured grids. pages Proceedings of the 4th international conference on computational fluid dynamics, Ghent, Belgium, 2006.

[61] S. Kang. Characteristics of flow over two circular cylinders in a side- by-side arrangement at low reynolds numbers. *Physics of Fluids*, 15:2486–2498, 2003.

[62] R. Kannan, Y. Sun, and Z. J. Wang. A study of viscous flux formulations for an implicit p-multigrid spectral volume navier stokes solver. 2008. AIAA Paper 2008-783.

[63] S. Kawai and S.K. Lele. Localized artificial diffusivity scheme for discontinuity capturing on curvilinear meshes. *Journal of Computational Physics*, 227:9498–9526, 2008.

[64] S. Kawai, S. K. Shankar, and S. K. Lele. Assessment of localized artificial diffusivity scheme for large-eddy simulation of compressible turbulent flows. *Journal of Computational Physics*, 229:1739–1762, 2010.

[65] D. A. Kopriva. A conservative staggered-grid chebyshev multidomain method for compressible flows. ii semi-structured method. *Journal of Computational Physics*, 128:475–488, 1996.

[66] D. A. Kopriva. A staggered-grid multidomain spectral method for the compressible navier-stokes equations. *Journal of Computational Physics*, 143:125–158, 1998.

[67] D. A. Kopriva and J. H. Kolias. A conservative staggered-grid chebyshev multidomain method for compressible flows. *Journal of Computational Physics*, 125:244–261, 1996.

[68] K. Z. Korczak and A. T. Patera. An isoparametric spectral element method for solution of the navier-stokes equations in complex geometry. *Journal of Computational Physics*, 62:361–382, 1984.

[69] S. K. Lele. Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics*, 103:16–42, 1992.

[70] C. Liang, R. Kannan, and Z.J. Wang. A p-multigrid spectral difference method with explicit and implicit smoothers on unstructured triangular grids. *Computers and Fluids*, 38:254–265, 2009.

[71] C. Liang, K. Ou, S. Premasuthan, A. Jameson, and Z. J. Wang. High-order accurate simulations of unsteady flow past plunging and pitching airfoils. *Computers and Fluids*. submitted.

[72] C. Liang, S. Premasuthan, and A. Jameson. High-order accurate simulation of flow past two side-by-side cylinders with spectral difference method. *Journal of Computers and Structures*, 87:812–817, 2009.

[73] X. D. Liu, S. Osher, and T. Chan. Weighted essentially nonoscillatory schemes. *Journal of Computational Physics*, 115:200–212, 1994.

[74] Y. Liu, M. Vinokur, and Z. J. Wang. Spectral difference method for unstructured grids i: Basic formulation. *Journal of Computational Physics*, 216:780–801, 2006.

[75] Y. Liu, M. Vinokur, and Z. J. Wang. Spectral (finite) volume method for conservation laws on unstructured grids v: Extension to three-dimensional systems. *Journal of Computational Physics*, 212:454–472, 2006.

[76] R. Lohner. Adaptive remeshing for transient problems with moving bodies. 1988. AIAA Paper 88-3737.

[77] H. Luo, J. D. Baum, and R. Lohner. A p-multigrid discontinuous galerkin method for the euler equations on unstructured grids. *Journal of Computational Physics*, 211:767–783, 2006.

[78] Y. Maday, C. Mavriplis, and A. T. Patera. Non-conforming mortar element methods: Application to spectral discretizations. In *Domain Decomposition Methods*.

[79] Y. Maday and R. Munoz. Spectral element multigrid part 2: theoretical justification. 1988. Technical Report ICASE 88-73.

[80] A. Mani, J. Larsson, and P. Moin. Suitability of artificial bulk viscosity for large-eddy simulation of turbulent flows with shocks. *Journal of Computational Physics*, 228:7368–7374, 2009.

[81] D.J. Mavripilis and A. Jameson. *Multigrid Solution of the Euler Equations on Unstructured and Adaptive Meshes, Lecture Notes in Pure and Applied Mathematics*, pages 413–430. Dekker, 1988.

[82] D. J. Mavriplis. Multigrid strategies for viscous flow solvers on anisotropic unstructured meshes. *Journal of Computational Physics*, 145:141–165, 1998.

[83] G. May, F. Iacono, and A. Jameson. Efficient algorithms for high-order discretizations of the euler and navier-stokes equations. 2009. AIAA Paper 2009-0182.

[84] G. May and A. Jameson. A spectral difference method for the euler and navier-stokes equations. 2006. AIAA Paper 2006-304.

[85] J. R. Meneghini, F. Saltara, C. L. R. Siqueira, and J. A. Ferrari. Numerical simulation of flow interference between two circular cylinders in tandem and side-by-side arrangements. *Fluids and Structures*, 15:327–350, 2001.

[86] C. Michalak and C. Ollivier-Gooch. Unstructured high-order accurate finite-volume solutions of the navier-stokes equations. 2009. AIAA Paper 2009-954.

[87] R. H. Ni. A multi-grid scheme for solving the euler equations. *AIAA Journal*, 20:1565–1571, 1981.

[88] J. T. Oden, I. Babuska, and C. E. Baumann. A discontinuous hp-finite element method for diffusion problems. *Journal of Computational Physics*, 146:491–519, 1998.

[89] C. F. Ollivier-Gooch. Quasi-eno schemes for unstructured meshes based on unlimited data-dependent least-squares reconstruction. *Journal of Computational Physics*, 133:6–17, 1997.

[90] K. Ou, C. Liang, and A. Jameson. A high-order spectral difference method for the navier-stokes equations on unstructured moving deformable grids. 2010. AIAA Paper 2010-541.

[91] M. Parsani, K. Van den Abeele, , and C. Lacor. Implicit lu-sgs time integration algorithm for high-order spectral volume method with pmultigrid strategy. In *West-East High Speed Flow Field Conference, Moscow, Russia*, 2007.

[92] A. T. Patera. A spectral element method for fluid dynamics: laminar flow in a channel expansion.

[93] J. Peraire, J. Peiro, L. Formaggia, K. Morgan, and O. Zienkiewicz. Finite element euler computations in three dimensions. 1988. AIAA Paper 88-0032.

[94] J. Peraire and P. O. Persson. The compact discontinuous galerkin (cdg) method for elliptic problems. *SIAM Journal on Scientific Computing*, 30:1806–1824, 2008.

[95] P. O. Persson and J. Peraire. Sub-cell shock capturing for discontinuous galerkin methods. 2006. AIAA Paper 2006-112.

[96] S. Premasuthan, C. Liang, and A. Jameson. A spectral difference method for viscous compressible flows with shocks. 2009. AIAA Paper 2009-3785.

[97] S. Premasuthan, C. Liang, and A. Jameson. Computation of flows with shocks using spectral difference scheme with artificial viscosity. 2010. AIAA Paper 2010-1449.

[98] S. Premasuthan, C. Liang, A. Jameson, and Z. J. Wang. A p-multigrid spectral difference method for viscous compressible flow using 2d quadrilateral meshes. 2009. AIAA Paper 2009-0950.

[99] P. Rasetarinera and M. Y. Hussaini. An efficient implicit discontinuous galerkin method. *Journal of Computational Physics*, 172:718–738, 2001.

[100] W. Reed and T. Hill. Triangular mesh methods for the neutron transport equation. 1973. Technical Report LA-UR 73-479, Los Alamos National Laboratory.

[101] R.C. Ripley, F.S. Lien, and M.M. Yoyanivich. Adaptive unstructured mesh refinement of supersonic channel flows. *International Journal of Computational Fluid Dynamics*, 18:189–198, 2004.

[102] P. L. Roe. Approximate riemann solvers, parameter vectors and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[103] E. Ronquist and A. Patera. Spectral element multigrid i. formulation and numerical results. *Journal of Scientific Computing*, 2:389–406, 1987.

[104] V.V. Rusanov. Calculation of interaction of non-steady shock waves with obstacles. *Journal of Computational and Mathematical Physics USSR*, 1:267–279, 1961.

[105] B. Sharman, F. S. Lien, L. Davidson, and C. Norberg. Numerical predictions of low reynolds number flows over two tandem circular cylinders. *International Journal for Numerical Methods in Fluids*, 47:423–447, 2005.

[106] C. W. Shu and S. S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77:439–471, 1988.

[107] G. A. Sod. A survey of several finite difference methods for sytems on non-linear hyperbolic conservation laws. *Journal of Computational Physics*, 27:1–31, 1981.

[108] R. J. Spiteri and S. J. Ruuth. A new class of optimal high-order strong-stability-preserving time discretization methods. *SIAM Journal of Numerical Analysis*, 40:469–491, 2002.

[109] Y. Sun and Z. J. Wang. Evaluation of discontinuous galerkin and spectral volume methods for scalar and system conservation laws on unstructured grids. *International Journal of Numerical Methods in Fluids*, 45:819–838, 2004.

[110] Y. Sun, Z. J. Wang, and Y. Liu. Spectral (finite) volume method for conservation laws on unstructured grids vi: Extension to viscous flow. *Journal of Computational Physics*, 215:41–58, 2006.

[111] Y. Sun, Z. J. Wang, and Y. Liu. High-order multidomain spectral difference method for the navier-stokes equations on unstructured hexahedral grids. *Communications in Computational Physics*, 2:310–333, 2007.

[112] Y. Sun, Z. J. Wang, and Y. Liu. Efficient implicit non-linear lu-sgs approach for compressible flow computation using high-order spectral difference method. *Communications in Computational Physics*, 5:760–778, 2009.

[113] Y. Sun, Z. J. Wang, Y. Liu, and C. L. Chen. Efficient implicit lusgs algorithm for high-order spectral difference method on unstructured hexahedral grids. 2007. AIAA Paper 2007-313.

[114] K. van den Abeele, T. Broeckhoven, and C. Lacor. Dispersion and dissipation properties of the 1d spectral volume method and application to a p-multigrid algorithm. *Journal of Computational Physics*, 224:616–636, 2007.

[115] K. van den Abeele, C. Lacor, and Z. J.Wang. On the stability and accuracy of the spectral difference method. *Journal of Scientific Computing*, 37:162–188, 2008.

[116] K. van den Abeele, C. Lacor, and Z. J. Wang. On the connection between the spectral volume and the spectral difference method. *Journal of Computational Physics*, 227:877–885, 2007.

[117] K. van den Abeele, M. Parsani, and C. Lacor. An implicit spectral difference navier-stokes solver for unstructured hexahedral grids. 2009. AIAA Paper 2009-0181.

[118] K. van den Abeele, M. Parsani, C. Lacor, and T. Quintino. A spectral volume navier-stokes solver on unstructured tetrahedral grids. In *Proceedings of 8th World Congress on Computational Mechanics (WCCM8)/5th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS 2008), Venice, Italy*, 2008.

[119] B. van Leer. Towards the ultimate conservative difference scheme. iv. a new approach to numerical convection. *Journal of Computational Physics*, 23:276–299, 1977.

[120] J.C. Vassberg and A. Jameson. In pursuit of grid convergence, part i: Two-dimensional euler solutions. 2009. AIAA Paper 2009-4114.

[121] D. A. Venditti and D. L. Darmofal. Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows. *Journal of Computational Physics*, 187:22–46, 2003.

[122] V. Venkatakrishnan, S. R. Allmaras, D. S. Kamenetskii, and F. T. Johnson. Higher order schemes for the compressible navier-stokes equations. 2003. AIAA Paper 2003-3987.

[123] M. R. Visbal and D. V. Gaitonde. On the use of higher-order finite-difference schemes of curvilinear and deforming meshes. *Journal of Computational Physics*, 181:155–185, 2002.

[124] J. von Neumann and R. Richtmyer. A method for the numerical calculation of hydrodynamic shocks. *Journal of Applied Physics*, 21:232–237, 1950.

[125] L. Wang and D.J. Mavripilis. Adjoint-based h-p adaptive discontinuous galerkin methods for the compressible euler equations. 2009. AIAA Paper 2009-952.

[126] Z. J. Wang. Spectral (finite) volume method for conservation laws on unstructured grids: Basic formulation. *Journal of Computational Physics*, 178:210–251, 2002.

[127] Z. J. Wang and H.Gao. A unifying lifting collocation penalty formulation including the discontinuous galerkin, spectral volume/difference methods for conservation laws on mixed grids. *Journal of Computational Physics*, 228:8161–8186, 2009.

[128] Z. J. Wang and Y. Liu. Spectral (finite) volume method for conservation laws on unstructured grids ii: Extension to two-dimensional scalar equation. *Journal of Computational Physics*, 179:665–697, 2002.

[129] Z. J. Wang and Y. Liu. Spectral (finite) volume method for conservation laws on unstructured grids iii: One dimensional systems and partition optimization. *Journal of Scientific Computing*, 20:137–157, 2004.

[130] Z. J. Wang and Y. Liu. Extension of the spectral volume method to high-order boundary representation. *Journal of Computational Physics*, 211:154–178, 2006.

[131] Z. J. Wang, Y. Liu, G. May, and A. Jameson. Spectral difference method for unstructured grids ii: Extension to the euler equations. *Journal of Scientific Computing*, 32:45–71, 2006.

[132] Z. J. Wang, L. Zhang, and Y. Liu. Spectral (finite) volume method for conservation laws on unstructured grids iv: Extension to two-dimensional euler equations. *Journal of Computational Physics*, 194:716–741, 2004.

[133] C. H. K. Williamson. Oblique and parallel modes of vortex shedding in the wake of a circular cylinder at low reynolds numbers. *Journal of Fluid Mechanics*, 206:579–627, 1989.

[134] M. Zhang and C. W. Shu. An analysis of and a comparison between the discontinuous galerkin and the spectral finite volume methods. *Journal of Computational Fluids*, 34:581–592, 2005.

[135] D.W. Zingg, S. De Rango, M. Nemec, and T. H. Pulliam. Comparison of several spatial discretizations for the navier-stokes equations. *Journal of Computational Physics*, 160:683–704, 2000.