Efficient Solution Techniques for Discontinuous Galerkin Discretizations of the Navier-Stokes Equations on Hybrid Anisotropic Meshes

Nicholas K. Burgess [*]Cristian R. Nastase [†]
and
Dimitri J. Mavriplis [‡]

Department of Mechanical Engineering, University of Wyoming, Laramie WY 82071, USA
and
Luigi Martinelli [§]
Department of Mechanical and Aerospace Engineering, Princeton University, Princeton NJ 08540, USA

The goal of this paper is to investigate and develop fast and robust solution techniques for high-order accurate Discontinuous Galerkin discretizations of non-linear systems of conservation laws on unstructured meshes. Previous work was focused on the development of hp-multigrid techniques for inviscid flows and the current work concentrates on the extension of these solvers to steady-state viscous flows including the effects of highly anisotropic hybrid meshes. Efficiency and robustness are improved through the use of mixed triangular and quadrilateral mesh elements, the formulation of local order-reduction techniques, the development of a line-implicit Jacobi smoother, and the implementation of a Newton-GMRES solution technique. The methodology is developed for the two- and three-dimensional Navier-Stokes equations on unstructured anisotropic grids, using linear multigrid schemes. Results are presented for a flat plate boundary layer and for flow over a NACA0012 airfoil and a two-element airfoil. Current results demonstrate convergence rates which are independent of the degree of mesh anisotropy, order of accuracy (p) of the discretization and level of mesh resolution (h). Additionally, preliminary results of on-going work for the extension to the Reynolds Averaged Navier-Stokes(RANS) equations, the development of a Gas-Kinetic (BGK) inter-cell flux function implementation, and the extension to three dimensions are given.

I. Introduction

Interest in the use of higher-order discretizations (higher than second-order) for industrial computational fluid dynamic problems, including aerodynamics, has become more widespread over the last several years. This is partly due to the difficulties encountered with traditional second-order accurate methods at reliably delivering consistently grid converged results and in quantifying the spatial discretization errors.[1,2] The asymptotic properties of higher-order methods makes them suitable for problems where high spatial accuracy is required, since for smooth solutions, spatial error is reduced ever more rapidly with increasing grid resolution at higher “p” orders of accuracy.

Discontinuous Galerkin (DG) methods are capable of generating high-order solutions to the Euler and Navier-Stokes equations and thus can deliver faster asymptotic error convergence rates compared to traditional second-order

[*] Ph.D. Candidate, Department of Mechanical Engineering, University of Wyoming
[†] Post Doctoral Research Associate, Department of Mechanical Engineering, University of Wyoming
[‡] Professor, Department of Mechanical Engineering, University of Wyoming
[§] Associate Professor, Department of Mechanical and Aerospace Engineering, Princeton University

finite-volume techniques. For increasingly high accuracy tolerances the use of high-order methods such as DG becomes more appealing. However, as the order of accuracy is increased the number of degrees of freedom(DoF) rises rapidly. For example second-order accurate ($p = 1$) DG discretizations for the two-dimensional Euler or Navier-Stokes equations have 12 DoF per element while a fourth-order accurate ($p = 3$) element has 40 DoF per element. Thus high-order methods have a significant cost associated with them and this cost scales with the order of the element.

As such the development of optimal, or near optimal solution strategies for higher-order discretizations, including steady-state solution methodologies, and implicit time-integration strategies, remains one of the key determining factors in devising higher-order methods which are not just competitive but superior to lower-order methods in overall accuracy and efficiency.

Recent work has examined the use of *hp*-multigrid methods, where convergence acceleration is achieved through the use of coarse levels constructed by reducing the order (p) of approximation of the discretization together with an agglomeration multigrid technique (h-multigrid) for Discontinuous Galerkin discretizations. This paper represents an extension to the Navier-Stokes equations of the previous work originally presented in references.[3,4] The extension includes the DG discretization of the viscous fluxes using the Symmetric Interior Penalty Method (SIP) together with the capability of using hybrid meshes on two-dimensional configurations. In order to maintain the convergence rate of the *hp*-multigrid solver on hybrid anisotropic meshes, a line creation algorithm and line-implicit Jacobi smoother have been devised and implemented . Additional solver efficiency is obtained through the implementation of a preconditioned Newton-GMRES Krylov method, which represents an alternative method for solving the linear system given by Newton's method. The methodology is developed for the two- and three-dimensional Navier-Stokes equations on unstructured anisotropic grids, using linear *hp*-multigrid schemes as preconditioners. While results presented here pertain only to steady-state problems, the methods are easily extended to time-dependent problems.[5] Extensions to the Reynolds-averaged Navier-Stokes equations, involving the solution of an additional turbulence modeling equation are investigated, as well as the use of alternate flux formulations derived from kinetic theory.

II. Governing Equations

The conservative form of the compressible Navier-Stokes equations describing the conservation of mass, momentum and total energy are given as

$$\frac{\partial \mathbf{U}(\mathbf{x},t)}{\partial t} + \nabla \cdot (\mathbf{F_c}(\mathbf{U}) - \mathbf{F_v}(\mathbf{U})) = 0 \tag{1}$$

subject to the appropriate boundary and initial conditions within a domain Ω. The state vector and flux vectors are explicitly given as

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E_t \end{pmatrix}, \quad \mathbf{F_c}^x = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(E_t + p) \end{pmatrix}, \quad \mathbf{F_c}^y = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ v(E_t + p) \end{pmatrix}, \quad \mathbf{F_c}^z = \begin{pmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ w(E_t + p) \end{pmatrix},$$

$$\mathbf{F_v}^x = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + k\frac{\partial T}{\partial x} \end{pmatrix}, \quad \mathbf{F_v}^y = \begin{pmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + k\frac{\partial T}{\partial y} \end{pmatrix}, \tag{2}$$

$$\mathbf{F_v}^z = \begin{pmatrix} 0 \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ u\tau_{zx} + v\tau_{zy} + w\tau_{zz} + k\frac{\partial T}{\partial z} \end{pmatrix},$$

where ρ is fluid density, (u,v,w) are the cartesian velocity components, p is the fluid pressure, E_t is the total energy, T is the fluid temperature, and τ_{ij} is the fluid viscous stress tensor. Assuming a Newtonian fluid the viscous stress tensor takes the form(with $\mathbf{x} = (x,y,z)$):

$$\begin{aligned} &\tau_{ij} = 2\mu S_{ij} \\ &S_{ij} = \frac{1}{2}\left(\frac{\partial \mathbf{u}_i}{\partial \mathbf{x}_j} + \frac{\partial \mathbf{u}_j}{\partial \mathbf{x}_i}\right) - \frac{1}{3}\frac{\partial \mathbf{u}_k}{\partial \mathbf{x}_k}\delta_{ij} \\ &\text{for } i = 1..3, j = 1..3 \end{aligned} \tag{3}$$

where μ is the fluid viscosity obtained via Sutherland's law. Furthermore, using the ideal gas assumption the pressure is given as:

$$p = (\gamma - 1)\left[E_t - \frac{1}{2}\rho\left(u^2 + v^2 + w^2\right)\right] \tag{4}$$

where $\gamma = 1.4$ is the ratio of specific heats.

III. Spatial Discretization

The computational domain Ω is partitioned into a set of non-overlapping elements and within each element the solution is approximated by a truncated polynomial expansion

$$\mathbf{U}(\mathbf{x},t) \approx \mathbf{U}_p(\mathbf{x},t) = \sum_{j=1}^{M} \mathbf{u}_j(t)\phi_j(\mathbf{x}) \tag{5}$$

where M is the number of modes defining the truncation level. The spatial discretization will be shown for a three dimensional domain. The simplification to two dimensions is a trivial process of removing the ρw state variable and the fluxes associated with it as well as the $\mathbf{F}^z$ and $\mathbf{F_v}^z$ flux vectors. The semi-discrete formulation (*i.e.* continuous in time) employs a local discontinuous Galerkin formulation[6–9] in spatial variables within each element Ω_k. The weak formulation for equation (1) is obtained by minimizing the residual with respect to the expansion function in an integral sense:

$$\int_{\Omega_k} \phi_i \left[\frac{\partial \mathbf{U}_p(\mathbf{x},t)}{\partial t} + \nabla\cdot\left[\mathbf{F_c}(\mathbf{U}_p) - \mathbf{F_v}(\mathbf{U}_p, \nabla\mathbf{U}_p)\right]\right]_k d\Omega_k = 0 \tag{6}$$

After integrating by parts the weak statement of the problem becomes:

$$\int_{\Omega_k} \phi_i \frac{\partial \mathbf{U}_p}{\partial t}\, d\Omega_k - \int_{\Omega_k} \nabla\phi_i \cdot \left[\mathbf{F_c}(\mathbf{U}_p) - \mathbf{F_v}(\mathbf{U}_p, \nabla\mathbf{U}_p)\right] d\Omega_k + \int_{\partial\Omega_k} [\![\phi_i]\!] \mathbf{F}^*_\mathbf{c}(\mathbf{U}_p)\cdot\mathbf{n} - \mathbf{F}^*_\mathbf{v}(\mathbf{U}_p, \nabla\mathbf{U}_p, \phi_i)\cdot\mathbf{n}\, d(\partial\Omega_k) = 0 \tag{7}$$

The local discontinuous Galerkin approach uses element-based basis functions that are discontinuous at the element interfaces, resulting in doubly valued solutions at these interfaces. For the convective terms the discontinuity is resolved using a monotone numerical flux, which provides inter-cell communication and boundary condition enforcement on the convective flux. The numerical flux, $\mathbf{F}^*_\mathbf{c}(\mathbf{U}_p)\cdot\mathbf{n}$, is obtained as a solution of a local one-dimensional Riemann problem and depends on the internal interface state, $\mathbf{U}^-_p$ and the adjacent element interface state, $\mathbf{U}^+_p$ as well as the orientation of the interface defined by the normal vector, $\mathbf{n}$. An approximate Riemann solver is used to compute the flux at inter-element boundaries. Current implementations include the flux difference splitting schemes of Rusanov,[10] Roe,[11] HLL[12] and HLLC.[13–15]

The numerical flux for the viscous term is obtained via the symmetric interior penalty method(SIP),[16,17] which seeks to penalize the solution for being discontinuous at the element interfaces. The SIP numerical flux is given as:

$$\mathbf{F}^*_\mathbf{v}(\mathbf{U}_p, \nabla\mathbf{U}_p, \phi_i)\cdot\mathbf{n} = [\![\phi_i]\!]\left\{\mathbf{F_v}(\mathbf{U}_p, \nabla\mathbf{U}_p)\right\}\cdot\mathbf{n} + [\![\mathbf{U}_p]\!]\left\{F_v(\mathbf{U}_p, \nabla\phi_i)\right\}\cdot\mathbf{n} - \nu\left\{\mathbf{G}_{ii}\right\}[\![\phi_i]\!][\![\mathbf{U}_p]\!] \tag{8}$$

where the $\{\}$ and $[\![\,]\!]$ are the average and jump operators respectively

$$\begin{aligned} \{\mathbf{a}\} &= \frac{\mathbf{a}^- + \mathbf{a}^+}{2} \\ [\![\mathbf{a}]\!] &= \left(\mathbf{a}^- - \mathbf{a}^+\right) \end{aligned} \tag{9}$$

where $\mathbf{a}$ is a scalar, vector or matrix and ν is the penalty parameter. The matrix $\mathbf{G}_{ij}$ is given such that.

$$\begin{aligned} \mathbf{F_v}^x &= \mathbf{G}_{11}\frac{\partial \mathbf{U}_p}{\partial x} + \mathbf{G}_{12}\frac{\partial \mathbf{U}_p}{\partial y} + \mathbf{G}_{13}\frac{\partial \mathbf{U}_p}{\partial z} \\ \mathbf{F_v}^y &= \mathbf{G}_{21}\frac{\partial \mathbf{U}_p}{\partial x} + \mathbf{G}_{22}\frac{\partial \mathbf{U}_p}{\partial y} + \mathbf{G}_{23}\frac{\partial \mathbf{U}_p}{\partial z} \\ \mathbf{F_v}^z &= \mathbf{G}_{31}\frac{\partial \mathbf{U}_p}{\partial x} + \mathbf{G}_{32}\frac{\partial \mathbf{U}_p}{\partial y} + \mathbf{G}_{33}\frac{\partial \mathbf{U}_p}{\partial z} \end{aligned} \tag{10}$$

The choice of penalty parameter can be rather ad hoc as it is only required to be “large enough” to stabilize the scheme. However, Shahbazi[16] derived an explicit expression for the penalty parameter for Poisson’s equation. A modified version of this expression given in reference[18] has been successfully implemented for this work. This explicit expression for the penalty parameter uses only the cell geometry and discretization order (p) to determine the value of the penalty parameter. Boundary conditions in the viscous flux are enforced by explicitly setting the first and second terms of equation (8) appropriately, while the third term uses the ghost values in the same fashion as the convective numerical flux.

The discrete equations are defined by the particular choice of the set of basis functions, $\{\phi_i, i = 1...M\}$. In this case the basis functions are defined as a set of hierarchical C^0 continuous *vertex*, *edge*, and *bubble* functions [19,20] that are based on variable weight Jacobi polynomials. The basis set is defined in the standard element $\widehat{\Omega}_k(\xi_j, j = 1\ldots3)$ spanning $\{-1 < \xi_j < 1\}$. A coordinate transformation, $\mathbf{x}_p = \mathbf{x}_p(\xi_1,\xi_2,\xi_3)$, between the master element and the physical element is required in order to compute the derivatives and integrals in the physical element $\Omega_k(x,y,z)$. For the superparametric elements used in the this work, the basis functions are expressed in the standard element i.e. as functions of ξ_1, ξ_2 and ξ_3, with the coordinate transformation and its Jacobian given by:

$$\mathbf{x}_p = \sum_{j=1}^{M_{p+1}} \widehat{\mathbf{x}}_j \phi_j(\xi_1,\xi_2,\xi_3), \qquad J_k(\xi_1,\xi_2,\xi_3) = \left| \frac{\partial(x,y,z)}{\partial(\xi_1,\xi_2,\xi_3)} \right| \tag{11}$$

where M_{p+1} is the number of modes in a $p+1$ discretization. Although for the case of a linear mapping and a triangular element the transform Jacobian J_k is a constant, this is not the case for linear quadrilaterals. With this in mind the solution expansion and weak statements become

$$\mathbf{U}_p(\xi,t) = \sum_{j=1}^{M} \widehat{\mathbf{u}}_j(t)\phi_j(\xi_1,\xi_2,\xi_3) \tag{12}$$

$$\int_{\widehat{\Omega}_k} \phi_i \frac{\partial \mathbf{U}_p}{\partial t} J_k d\widehat{\Omega}_k - \int_{\widehat{\Omega}_k} \nabla\phi_i \cdot [\mathbf{F_c}(\mathbf{U}_p) - \mathbf{F_v}(\mathbf{U}_p,\nabla\mathbf{U}_p)]\, J_k d\widehat{\Omega}_k + \int_{\partial\widehat{\Omega}_k} [[\![\phi_i]\!]\mathbf{F}^*_{\mathbf{c}}(\mathbf{U}_p)\cdot\mathbf{n} - \mathbf{F}^*_{\mathbf{v}}(\mathbf{U}_p,\nabla\mathbf{U}_p,\phi_i)\cdot\mathbf{n}]\, J_k d(\partial\widehat{\Omega}_k) = 0 \tag{13}$$

This set of equations is solved in modal space and the integrals are evaluated using Gaussian quadrature rules .[19,21,22] These quadrature rules require the projection of the solution from the modal space to the quadrature points. To preserve the $p+1$ accuracy of the DG scheme the volume integrals are computed using a rule that integrates a polynomial of degree $2p$ exactly. To the same end, surface integrations are carried out with a rule that integrates a polynomial of $2p+1$[23] exactly.

III.A. Mixed Element Meshes

Often in the finite-volume literature, mixed-element meshes are employed to improve the accuracy of gradient reconstruction in the boundary layer.[24] This is not the case in DG methods as gradients are computed via the derivatives of the basis functions and do not rely on the neighbor stencil (they do however rely on the transformation metrics). However, the use of quadrilateral elements for DG discretizations can still be beneficial since they enable equivalent accuracy with fewer elements (a single quadrilateral can replace two self-similar triangles).

Additionally, quadrilateral elements have been found to be more flexible for use with curved geometries in anisotropic regions of the mesh. In this work, we use super-parametric elements, where the element boundaries are mapped to order $p+1$ for a solution order of p. In the case of anisotropic meshes this often necessitates the curving of interior elements to avoid inconsistent mesh cross-overs. A variety of strategies exist for this purpose. For example references[25,26] have used elasticity-based node movement schemes. These methods have the advantage that they can produce highly stretched meshes with positive element transform Jacobians everywhere. On the other hand they lead to the entire mesh being mapped to higher-order. This is unnecessary as once the mesh has become isotropic and does not touch a curved boundary, straight-sided (linear) mappings are sufficient. As such in this work we have opted to use a simpler approach.

Here mappings of each of the curved boundary faces are generated. Then we make use of the already formed lines (for the line-implicit Jacobi smoother described subsequently in Section V.B) to copy this curvature straight up the line away from the surface. This approach ensures that only a small subset of the mesh is mapped to higher than $p=1$, saving computational cost when computing the transformation quantities. It does however have a disadvantage in that the element Jacobian can become negative, especially for triangular meshes, which is the result of edge-crossing for highly curved triangles. It is for this reason that quadrilaterals are employed in the boundary layer. These elements can tolerate much higher curvature and aspect ratios without negative transform Jacobians. In fact the authors have yet to produce a mesh where this has happened. Employing mixed element meshes for DG has allowed us to minimize transformation computational cost while allowing us to use very highly curved high aspect-ratio elements.

IV. The Implicit Steady State Solver

By neglecting the temporal derivative in equation (13) the steady-state system of equations corresponding to the last two spatial integrals is obtained. This may be written symbolically as:

$$\mathbf{R}(\mathbf{U}_p) = 0 \tag{14}$$

where $\mathbf{R}(\mathbf{U}_p)$ is the non-linear spatial residual. This set of non-linear equations is solved using Newton's method:

$$\begin{aligned} \left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}_p}\right]^n \Delta\mathbf{U}_p^{n+1} &= -\mathbf{R}(\mathbf{U}_p^n) \\ \mathbf{U}_p^{n+1} &= \mathbf{U}_p^n + \Delta\mathbf{U}_p^{n+1} \end{aligned} \tag{15}$$

It is well known that Newton's method will diverge if the initial guess is too far from the true solution. Thus the flux Jacobian matrix is augmented with a damping term to increase robustness. This new damped Newton iteration is given as:

$$\begin{aligned} \left[\frac{|\Omega|\,[I]}{\Delta t_n} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}_p}\right]^n \Delta\mathbf{U}_p^{n+1} &= -\mathbf{R}(\mathbf{U}_p^n) \\ \mathbf{U}_p^{n+1} &= \mathbf{U}_p^n + \Delta\mathbf{U}_p^{n+1} \end{aligned} \tag{16}$$

where Δt_n is the maximum stable timestep based on a p = 0 discretization and

$$\begin{aligned} &CFL = min\left(CFL_{min}\left(CFL_s\right)^n, CFL_{max}\right) \\ &\Delta t_{nk} = CFL\Delta t_{min} \end{aligned} \tag{17}$$

where the CFL_{min}, CFL_s and CFL_{max} are defined by the user and Δt_{min} is the ratio element perimeter to average maximum convective eigenvalue in the cell k. Due to the block-sparse nature and size of the matrix an iterative method will be used to solve the linear system arising from the Newton's method. Nastase and Mavriplis [3] review some approximations and splittings of the flux Jacobian. Here we employ an additional splitting namely the line-implicit Jacobi splitting. Consider the linear system $[A]x = b$ and let $[A]$ be split as $[A] = [M] + [N]$. Thus for iteration k one can write the update to x as

$$x^{k+1} = (1-\omega)x^k + \omega [M]^{-1} \left(b - [N]x^k \right) \tag{18}$$

where ω is an under relaxation factor $\omega \in (0,1]$. The forms of $[M]$ and $[N]$ give different simple iterative methods. In this work two splittings are considered. In the first, $[M]$ is taken as the block diagonal of the full Jacobian matrix and $[N]$ corresponds to all the block off-diagonals, which yields the so-called linearized element Jacobi (LEJ)[3] scheme. The other splitting yields the line-implicit Jacobi scheme where $[M]$ is now taken as the part of the flux Jacobian corresponding to the diagonal and off-diagonal blocks contained in a set of lines drawn through the highly stretched anisotropic regions of the mesh, and $[N]$ represents all remaining off-diagonal block elements. Note that in regions of isotropic cells (lines containing a single element) this splitting reverts to the LEJ splitting since the line length reduces to a single cell. Either of these splittings can be used as a solver by itself. However reference [18] has shown that they are better used as smoothers for multigrid methods, as preconditioners, or as part of preconditioners for Krylov subspace methods (GMRES).

V. Multigrid Methods

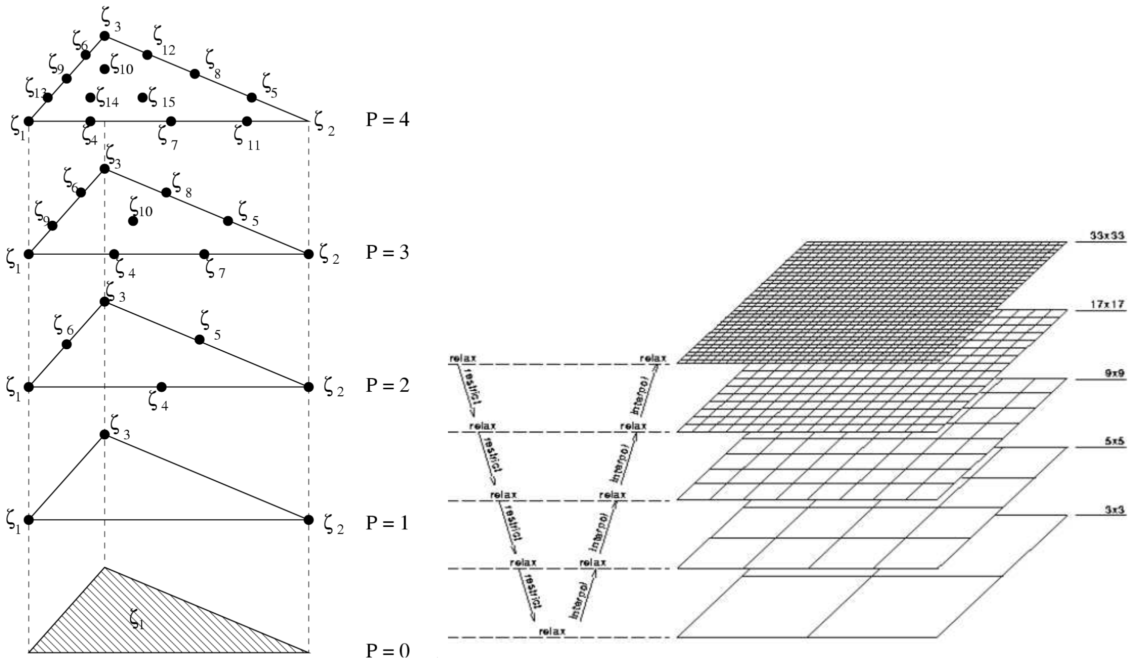
Multigrid methods are known as efficient techniques for accelerating convergence to steady-state for both linear and non-linear problems,[27,28] and can be applied with a suitable existing relaxation technique. The rapid convergence property relies on an efficient reduction of the solution error on a nested sequence of coarse grids.

V.A. The *hp*-Multigrid Approach

The spectral multigrid approach[29,30] is based on the same concepts as a traditional *h*-multigrid method, but makes use of "coarser" levels which are constructed by reducing the order of accuracy of the discretization, rather than using physically coarser grids with fewer elements. Thus, all grid levels contain the same number of elements, which alleviates the need to perform complex interpolation between grid levels and/or to implement agglomeration-type procedures.[28] Furthermore, the formulation of the interpolation operators, between fine and coarse grid levels, is greatly simplified when a hierarchical basis set is employed for the solution approximation. The main advantage is due to the fact that the lower-order basis functions are a subset of the higher-order basis (*i.e.* hierarchical) and the *restriction* and *prolongation* operators become simple projection operators into a lower- and higher-order space, respectively.[31] Therefore their formulation is obtained by a simple deletion or augmentation of the basis set. The *restriction* from fine level to coarse level is obtained by disregarding the higher-order modal coefficients and transferring the values of the low-order modal coefficients exactly. Similarly, the *prolongation* from coarse to fine levels is obtained by setting the high-order modes to zero and injecting the values of the low-order coefficients exactly. Figure 1(a) shows various *p*-levels for a reference triangular element.

Multigrid strategies are based on a recursive application of a two-level solution mechanism, where the second (coarser) grid is solved exactly, and used to accelerate the solution on the finer grid.[27] Because the exact solution of the coarse grid problem at each multigrid cycle is most often prohibitively expensive, the recursive application of multigrid to solve the coarse grid problem offers the preferred approach for minimizing the computational cost of the multigrid cycle, thus resulting in a complete sequence of coarser grids. For spectral (*p*)-multigrid methods, the recursive application of lower-order discretizations ends with the $p = 0$ discretization on the same grid as the fine level problem.

For relatively fine meshes, the (exact) solution of this $p = 0$ problem at each multigrid cycle can become expensive, and may impede the *h*-independence property of the multigrid strategy. The $p = 0$ problem can either be solved approximately by employing the same number of smoothing cycles on this level as on the finer p levels, or the $p = 0$ problem can be solved more accurately by performing a larger number of smoothing cycles at each visit to this coarsest level. In either case, the convergence efficiency will be compromised, either due to inadequate coarse level convergence, or to excessive coarse level solution cost. An alternative is to employ an *h*-multigrid procedure to solve the coarse level problem at each multigrid cycle. In this scenario, the p-multigrid scheme reverts to an agglomeration



(a) *p*-Multigrid

(b) *h*-Multigrid

Figure 1. Illustration of *hp*-Multigrid Levels

multigrid scheme once the $p = 0$ level has been reached, making use of a complete sequence of physically coarser agglomerated grids, thus the designation *hp*-multigrid. Agglomeration multigrid methods make use of an automatically generated sequence of coarser level meshes, formed by merging together neighboring fine grid elements, using a graph algorithm. Figure 1(b) shows a typical agglomeration multigrid configuration. First-order accurate ($p = 0$) agglomeration multigrid methods for unstructured meshes are well established and deliver near optimal grid independent convergence rates.[32]

The *hp*-multigrid procedure[3,4] has been shown to result in an *h*- and *p*-independent solution strategy for high-order accurate discontinuous Galerkin discretizations of the Euler equations, in both two- and three-dimensions. For robustness it is important to augment the resulting multi-level *hp*-multigrid scheme with a full multigrid (FMG) technique, in order to provide a good initial guess for the fine level problem. Moreover, the use of FMG is critically important in the case of the linear multigrid scheme for it is known that the Newton iteration will diverge if the initial guess is not close enough to the final solution. In our *hp*-multigrid approach, the solution process begins at the coarsest grid level ($p = 0$), using all the *h*-levels available, and ends at the fine level where all the *p*- and *h*-levels are used to advance the solution to the desired accuracy, as depicted in Figure 2. Alternatively the FMG strategy can be initiated at the coarsest *h*-level.

V.B. Line-Implicit Jacobi

In order to maintain the convergence rate of multigrid on anisotropic meshes a line-implicit smoother can be used.[33] With this in mind, a line creation algorithm and line-implicit Jacobi smoother have been devised and implemented to enable efficient solution techniques on anisotropic meshes.

Line creation is accomplished using a two pass approach. Since the anisotropic cells are used to capture boundary layers, anisotropic regions will be found attached to the boundary faces of the mesh that are solid surfaces. The line creation algorithm begins by considering all the boundary faces in the mesh attached to solid surfaces. Lines that originate from the solid surfaces are formed by taking each surface face and computing the corresponding normal vector. Then the angle between the surface normal and all other faces attached to the boundary element are computed. If any of the angles are less then 10 degrees then that face is added to the line and the neighboring element across that

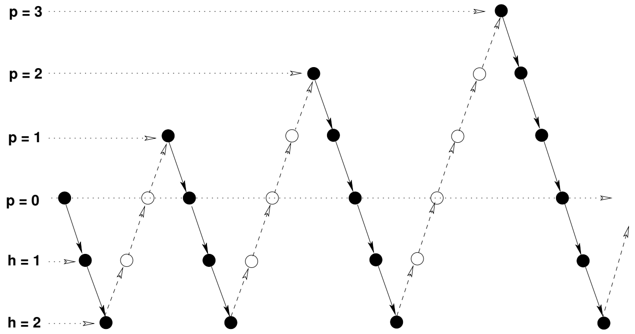


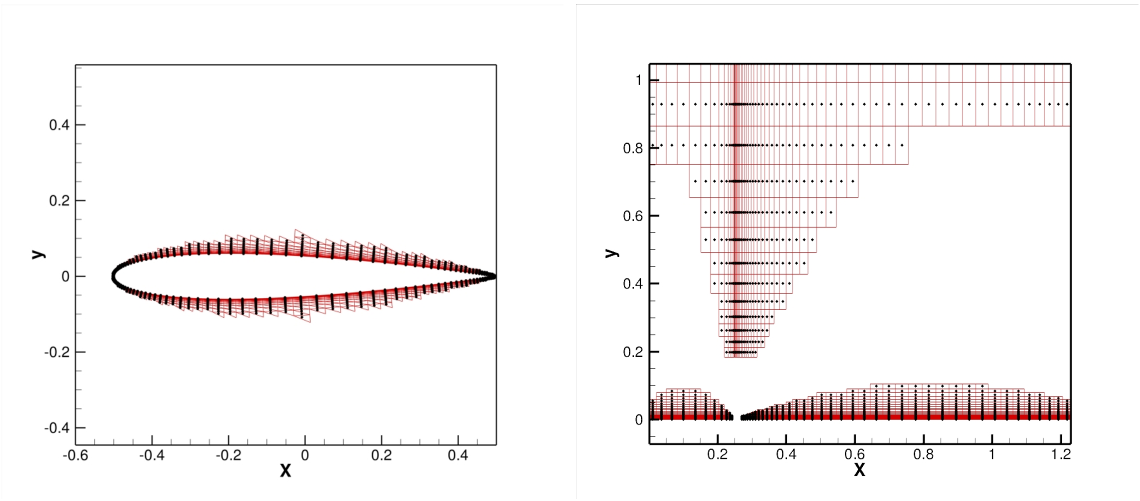
Figure 2. Illustration of Full hp-multigrid (FMG) levels for $p = 3$ and $h = 2$ (– restriction, - - prolongation, • smoothing, ○ update).

face becomes the element to be searched. The processes is repeated until all the current element's face normals make an angle with the surface normal greater than 10 degrees. This is done for each boundary face in the mesh that is a solid surface.

Once the lines attached to the solid surfaces are formed there can still be areas in the mesh that are highly anisotropic (*i,e.* wake regions). Lines in these regions are created using a weighted graph algorithm,[34] which has been modified slightly for cell-centered rather then node-centered discretizations. This algorithm assigns a weight to each graph edge (in this case a graph edge is a line connecting two cell-centers). This edge-weight is taken as the inverse of the graph edge length. The ratio of maximum to average edge-weight is precomputed for each cell in the mesh. The cells are then ordered according to this ratio and stored as a heap. The top element in the heap is chosen as the starting point for a line, provided it is not already part of a line. The most strongly connected neighbor is added to the line provided that it is not already part of a line and that the ratio of maximum to minimum connection strength is greater then β ($\beta = 3$ for all cases). The element that was just added to the line now becomes the current element. The line is terminated when the current element violates the threshold on β. As these lines are not attached to a surface the process is repeated for the original seed element with the second most strongly connected neighbor and proceeding as before adding to the same line. Figure 3 shows lines created around a NACA0012 airfoil and those created on a flat plate of zero thickness. These two examples demonstrate the different pieces of the above algorithm in isolation. The lines around the NACA0012 are created via the surface normal approach, while the lines for the quadrilateral flat plate mesh are created with the weighted graph algorithm. Note that the weighted graph algorithm has formed lines in the convective direction off the solid surface. This is due to the cell clustering at the plate leading edge. In fact if these lines are not created the convergence rate of the multigrid solver degrades significantly, showing that lines are needed in highly anisotropic regions of the mesh regardless of the physical phenomena present in those regions.

Once the lines have been created the line-implicit smoother is straight-forward to implement. Implicit lines form a block tridiagonal matrix for each line. Each row of this block tridiagonal matrix contains the block diagonal matrix of an element and at least one (usually two) block off-diagonal matrices. This is how $[M]$ and $[N]$ in equation (18) are defined. $[M]$ contains all the block matrices in a line and $[N]$ those block off-diagonal matrices that are not in a line. Using this matrix splitting the linear system is now solved using a block variant of the Thomas algorithm .[35]

The reason for using the line-implicit smoother is to ensure that the convergence rate observed for isotropic meshes is maintained for similar anisotropic meshes. To verify that this is the case, the line-smoother was implemented within a linear multigrid algorithm and used to solve Poisson's equation. Two meshes each of which contain N = 1102 elements were used. The first mesh is made of isotropic evenly spaced triangles and the second is made of anisotropic stretched triangles(maximum aspect ratio = 26912 : 1). The results of solving Poisson's equation with a p = 3 DG discretization using linear multigrid are shown in Table 1. These results very clearly show that without the line-implicit smoother the convergence rate drops significantly. On the other hand, if the line-implicit smoother is used the convergence rate is almost unchanged.



(a) NACA0012

(b) Flat plate

Figure 3. Illustration of lines used in line-implicit Jacobi smoother

Table 1. Comparison of convergence rates for Poisson problem,using N = 1102 elements with and without stretching for p = 3

Mesh/Smoother	Average rate
Isotropic/LEJ	.49
Anisotropic/LEJ	.98
Anisotropic/Implicit-line	.55

VI. Newton-GMRES

Krylov Subspace Methods represent an alternative technique for solving the linear system given by Newton's method. Since the linear system is non-symmetric, we rely on the Generalized Minimal Residual method (GMRES). The convergence of GMRES is often accelerated using a preconditioner. In this work we choose to use the previously described linear multigrid algorithm as the preconditioner. Reference [18] has shown this to be a nearly optimal method for isotropic meshes, and showed some promising preliminary results for triangular anisotropic meshes. In this work this method has been extended to hybrid anisotropic meshes.

We rewrite equation (16) as

$$[A]x = b \tag{19}$$

where $[A]$ is the flux Jacobian, x is the Newton update and b is minus the flow residual. GMRES seeks to minimize the L_2 norm of a linear system's residual($r = [A]x - b$) over the space $span\{r_0, [A]r_0, [A]^2 r_0, \ldots, A^{k-1} r_0\}$. There are many excellent texts on Krylov subspace methods [36–38] and we refer the reader to those for more details on the theory of Krylov subspace methods. For a right-preconditioned system one obtains the following

$$[A]\,[P]^{-1} y = b, \quad x = [P]^{-1} y \tag{20}$$

In this case the Krylov subspace is $span\{r_0, [A]\,[P]^{-1}\,r_0, ([A]\,[P]^{-1})^2 r_0, \ldots, ([A]\,[P]^{-1})^{k-1} r_0\}$. To compute the Krylov subspace basis a linear system of the form

$$[P]z = q \tag{21}$$

must be solved. This equation is solved approximately using a few cycles of multigrid. Using multigrid as the preconditioner yields an algorithm we denote as multigrid preconditioned GMRES (MGPC-GMRES). Note that the preconditioning matrix is the full flux Jacobian and is never fully inverted. Also note that GMRES requires additional storage for the Krylov basis. However, we expect the multigrid preconditioner to be very effective so the size of the Krylov basis will remain small. The details of the GMRES solver are given in algorithm 1.

Algorithm 1 :Multigrid preconditioned fGRMES algorithm

```
$N =$ # of Krylov vectors, $M =$ # of linear iterations
Given $x_1$, compute $r_1 = b - [A]x_1$
i = 1
while $i <= M$ .and. $\|r_i\| > tol$ do
    set $q_1 = \frac{r_1}{\|r_1\|}$
    set $\zeta(:) = 0$
    set $\zeta(1) = \|r_1\|$
    n = 1
    while $n <= N$ .and. $res > tol$ do
        Solve $[P]z_n = q_n$ using linear multigrid
        $[Z(:,n)]_i = z_n$
        $v = [A]z_n$
        % Compute new Krylov basis vector via Arnoldi proceedure
        for $j = 1,n$ do
            $h(j,n) = v \cdot q_j$
            $v = v - h(j,n)q_j$
        end for
        $h(n+1,n) = \|v\|$
        $q_{n+1} = v/h(n+1,n)$
        % Make the upper Hessenberg matrix upper triangular via givens rotations
        for $j = 1,n-1$ do
            $temp = cs(j)h(j,n) + sn(j)h(j+1,n)$
            $h(j+1,n) = -sn(j)h(j,n) + cs(j)h(j+1,n)$
            $h(j,n) = temp$
        end for
        % Compute the $n^{th}$ givens rotation matrix( a very standard approach given in)[39]
        call get_rot(h(n,n), h(n + 1, n), cs(n), sn(n))
        $h(n,n) = cs(n)h(n,n) + sn(n)h(n+1,n)$
        $temp = cs(n)zeta(n)$
        $\zeta(n+1) = -sn(n)zeta(n)$
        $\zeta(n) = temp$
        res = $\frac{|\zeta(n+1)|}{\|b\|}$
    end while
    solve $[h]y = \zeta$
    $x_{i+1} = x_i + [Z_i]y_i$
    % where the $n^{th}$ column of $[Z_i]$ is $z_n$
    $i = i+1$
    $r_i = b - [A]x_i$
end while
```

VII. Robustness Improvement

VII.A. Local Order Reduction

One of the principal concerns when developing DG solvers is the robustness of the solver. The authors and others[26] have noted that under mesh-resolved regions of the flow can adversely impact the convergence of the flow solver.

If regions of smooth extrema are under mesh-resolved the smooth extrema can become non-smooth extrema. When these extrema become non-smooth the DG discretization may produce unphysical oscillations which may lead to solver failure(as will be shown in Section: VIII.C).

A simple way of dealing with this problem is to simply refine the mesh in the under resolved region, thus causing the extrema to become smooth again. However, adaptive mesh refinement may not always be feasible, and it is desirable to have a robust solver capable of producing solutions even in the presence of under-resolved features. To decrease the oscillations that result from these under-resolved areas one can add additional diffusion to the equations to smooth out the solution. This can be done in a variety of ways. For example references[40,41] have explicitly added Laplacian type diffusion terms for shocks. Yet another approach is to recognize that as the jumps between elements increase so does the artificial diffusion associated with the upwinding. With this in mind we have employed a local element order-reduction technique based on the element resolution detector developed in reference.[40] By decreasing the order of the element we increase the inter-element jumps and thus the artificial diffusion of the flux function. The detector for an element k is given as

$$\begin{aligned} S_k &= \frac{\langle u_k - \bar{u}_k, u_k - \bar{u}_k \rangle}{\langle u_k, u_k \rangle} \\ u_k &= \sum_{j=1}^{M} \hat{u}_j \phi_j \\ \bar{u}_k &= \sum_{j=1}^{M_{p-1}} \hat{u}_j \phi_j \end{aligned} \tag{22}$$

with M_{p-1} the number of modes in a $p-1$ discretization and u is any quantity of interest in the flow. Density has proven to be a very reliable quantity to use for this purpose. The inner product here is the standard L_2 inner product over each element k.

$$\langle u, v \rangle = \int_{\widehat{\Omega_k}} u v J_k d\widehat{\Omega_k} \tag{23}$$

The value of S_k for each element k in the mesh is computed. Then if $log_{10}(S_k)$ is greater then -4 for an element, that element's order is reduced to $p_k = min(p_k - 1, 1)$ (here $p = 1$ is the floor to preserve all of the viscous terms). This approach was used here simply to test the concept of treating these kinds of under resolved regions as though they were flow discontinuities.

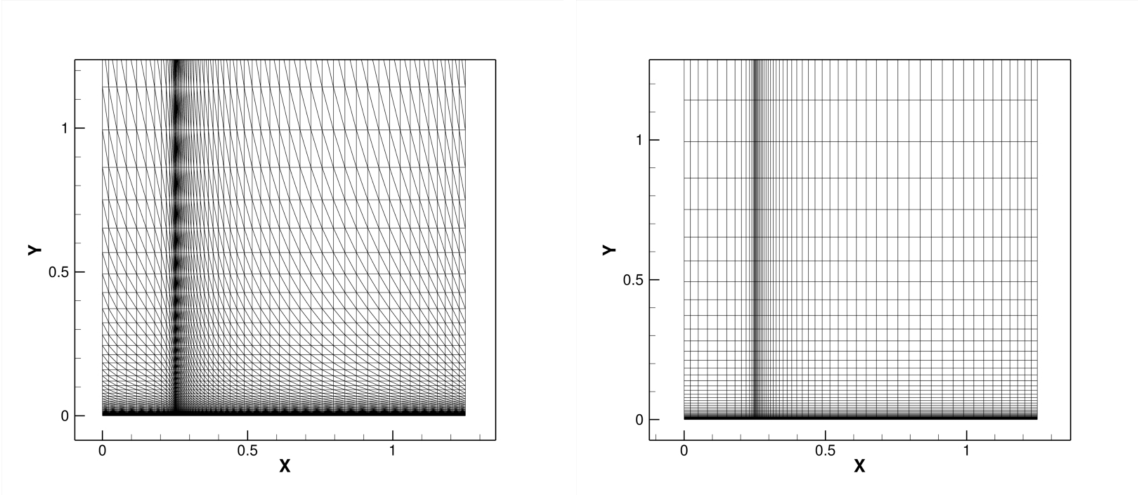
VIII. Numerical Results

The flow solver is validated and tested using three separate test cases, a laminar flat plate, a NACA0012 airfoil, and a two-element airfoil. The laminar flat plate test case is used to validate the Navier-Stokes terms in the flow solver. The MGPC-GMRES solver's efficiency is tested using a NACA0012 airfoil and a two-element airfoil. In addition preliminary Reynolds-averaged Navier-Stokes results using a zero pressure gradient flat plate and one-dimensional BGK flux results[42] are reported.

VIII.A. Laminar Flat-Plate

The laminar viscous solver is validated using a zero pressure gradient flat plate boundary layer at the following conditions $M_\infty = .1$, $\alpha = 0^o$, and $Re = 200{,}000$. The velocity profiles are compared to the standard Blasius solution. This flow been computed for p = 0 through p = 3 for both a triangular mesh and a quadrilateral mesh with N = 6372 and N = 3186 elements respectively. The triangular mesh is built from the quadrilateral mesh with each quadrilateral divided into two triangles. This was done to maintain the spacing normal to the wall and the cell height to length ratios between the two meshes. The meshes for this case are shown in Figures 4(a) and 4(b).

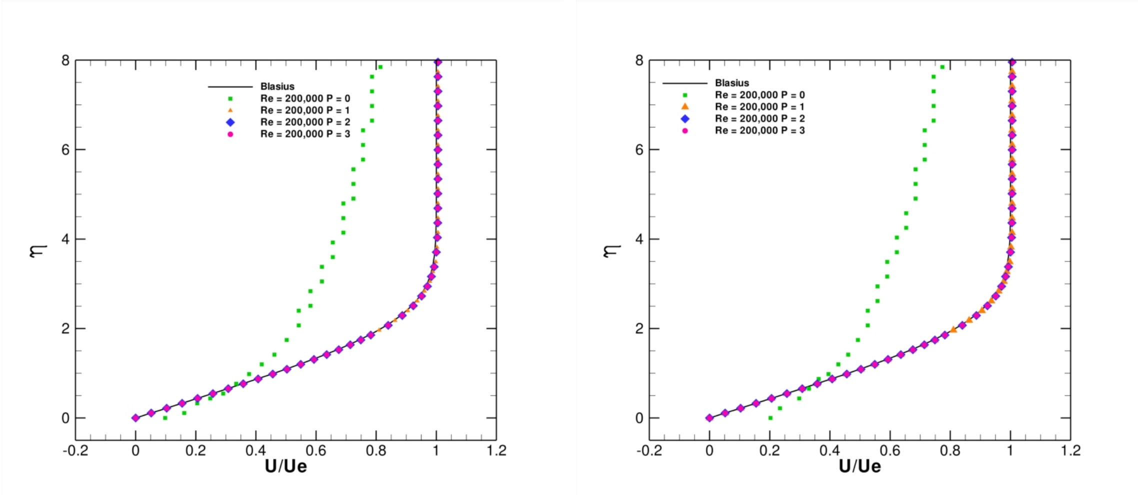
Figures 5(a) through 6(b) show the computed velocity profiles with the Blasius solution plotted as a reference. The computed velocity profiles agree very well for orders p = 1 and greater. For p = 0 the velocity profile is extremely inaccurate, this is due to the fact that the DG discretization of the viscous operator reduces to an inconsistent edge-only approximation for the viscous terms. We have included these results simply to show that p = 0 is not sufficient to fully capture the viscous operator. The u-velocity profile matches the Blasius profile for all orders. The v-velocity profile is the harder of the two profiles to capture accurately. The second-order result on the triangular mesh shows a deviation from the Blasius solution in this profile, but the higher-order results show improved agreement. Additionally note that above p = 1, there is essentially no difference in the agreement between the triangular and quadrilateral meshes. It should be noted that for p = 3 the quadrilateral case represents a 25% reduction in the number of DoFs vs the p = 3 solution on the triangular mesh. Therefore quadrilateral meshes are capable of delivering equivalent or better accuracy compared to self-similar triangular meshes at a lower computational cost.



(a) N = 6372 Triangles

(b) N = 3186 Quadrilaterals

Figure 4. Meshes used for computing the laminar flow past a flat plate



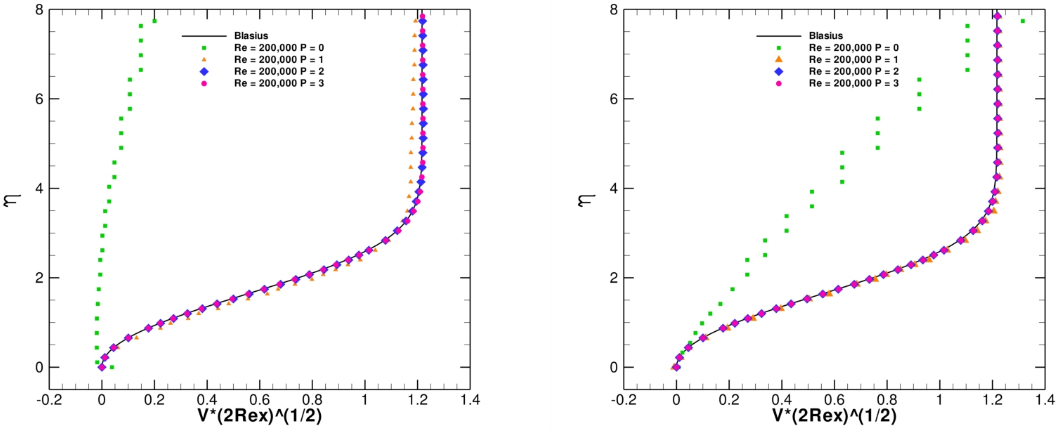
(a) Triangular Mesh

(b) Quadrilateral Mesh

Figure 5. Laminar flat plate u-velocity profile computed using DG for p = 0 to p = 3 compared to the Blasius u-velocity profile

VIII.B. NACA0012

The effectiveness and efficiency of the MGPC-GMRES algorithm is studied using two test cases. The first is a NACA0012 airfoil at $M_\infty = .5$, $\alpha = 0^o$, and $Re = 5,000$, which is used to verify the h-independence and p-independence



(a) Triangular mesh

(b) Quadrilateral mesh

Figure 6. Laminar flat plate v-velocity profile computed using DG for p = 0 to p = 3 compared to the Blasius v-velocity profile

of the solver for purely triangular meshes. In each case an iteration represents one Newton iteration. For each Newton iteration the MGPC-GRMES solver convergences the linear system obtained via Newton's method to the tolerance given by equation (24) or until 15 Krylov vectors are generated.

$$
\begin{aligned}
&tol = max\left(min\left(\frac{||\mathbf{b}||_{L_2}}{2.5^{(n-nc-p)}}, .1||\mathbf{b}||_{L_2}\right), 1.0e-14\right)\\
&nc = 8+2\,(p-1)
\end{aligned}
\tag{24}
$$

In equation (24) p is the order, **b** is the right-hand side of the linear system, and n is the Newton iteration number. For each Krylov vector the preconditioning system given by equation (21) was solved with four cycles of the linear multigrid solver.

Two purely triangular meshes are used for this case and contain N = 2250 and N = 7750 triangles, with a maximum aspect ratio of 82:1 and 238:1, and with average line lengths of 10 and 25 cells respectively. Due to the variance in aspect ratio we expect there to be a slight h-dependence as aspect-ratio can affect the stiffness of the problem. Figures 8(a) and 8(b) show the respective convergence rate obtained and demonstrate relatively h-independent and p-independent behavior. These figures demonstrate that the MGPC-GMRES solver gives nearly h-independent and p-independent results, despite the difference in aspect-ratio. To ensure that the above results are also valid for mixed-element anisotropic meshes, the NACA0012 test case was computed on a mixed-element anisotropic mesh containing N = 4964 elements (Figure 9(a)) with a maximum aspect-ratio of 65:1 for DG discretization orders p = 1 to p = 4. Figure 9(b) depicts the convergence history using this mixed-element mesh. This figure demonstrates that the MGPC-GMRES solver retains its p-independence on mixed-element anisotropic meshes.

VIII.C. Two-Element Airfoil

The second test case is a two-element airfoil depicted in Figure 10(a). The flow conditions for this case are $M_\infty = .3$, $\alpha = 0^o$, and $Re = 5,000$. Solutions for p = 1 through p = 4 have been computed on N = 7902 elements (5266 triangles and 2636 quadrilaterals) with a maximum aspect ratio of 265:1 and using 252 lines with an average length of 15 cells. This two-element airfoil test case has been used to compare the performance of linear multigrid vs. MGPC-GMRES. Additionally we make a CPU time comparison between the LEJ smoother and line-implicit Jacobi smoother in the context of the linear multigrid solver. This case also served as the local order-reduction test case. These solutions were generated in a manner different from the NACA0012 case. To ensure accurate CPU time comparisons both the linear multigrid and MGPC-GMRES solvers were run such that each Newton iteration consisted of ten multigrid cycles. For

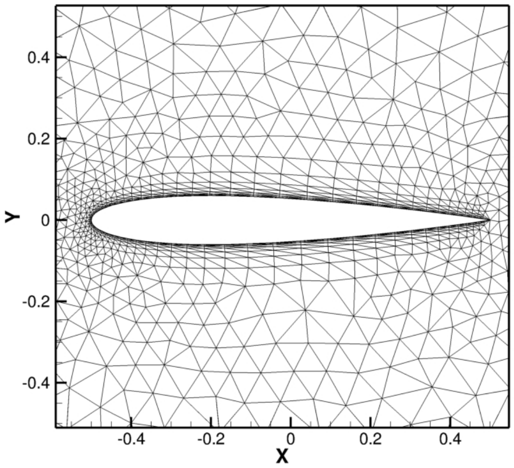
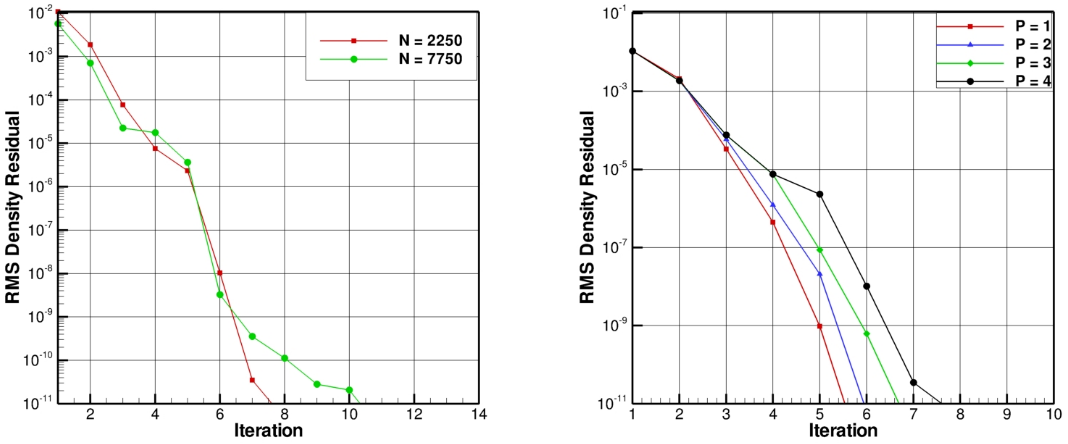


Figure 7. NACA0012 mesh with N = 2250 elements



(a) h-independence

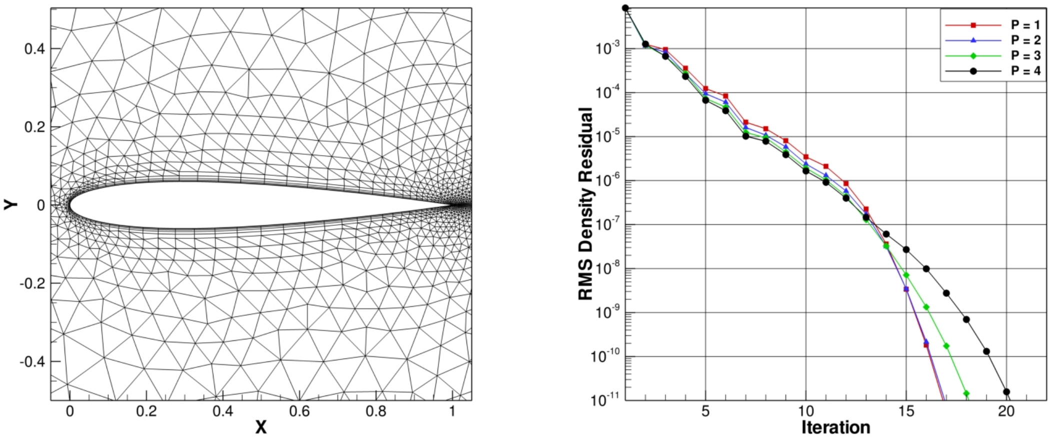
(b) p-independence

Figure 8. MGPC-GMRES convergence rates for the solution of the NACA0012 case on two meshes with N = 2250 and N = 7750 elements and for DG discretization orders p = 1 to p = 4

the linear multigrid solver this constituted ten multigrid cycles while for the MGPC-GMRES solver this constituted five Krylov vectors with two cycles of multigrid for preconditioning each Krylov vector.

VIII.C.1. Solver Comparison

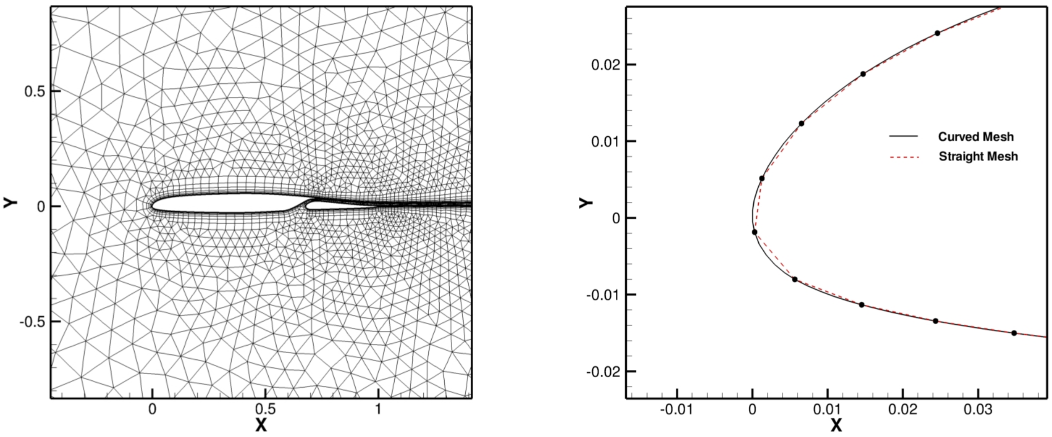
First let us consider the p-independence of the solvers. Figure 11(a) shows the convergence of such a solver for p = 1 through p = 4. The number of iterations required to solve p = 2 and higher is nearly twice that of p = 1, indicating a relatively strong p-dependence between p = 1 and higher-order ($p \geq 2$) solutions. The same problem has been solved using MGPC-GMRES and the convergence is depicted in Figure 11(b). The p-independence for the MGPC-GMRES solver shows a slight improvement over that of the linear multigrid solver because the convergence history of the



(a) Mixed-element mesh with N = 4964 elements

(b) p-independence

Figure 9. Mixed-element mesh containing N = 4964 elements (220 quadrilaterals and 4744 triangles) and MGPC-GMRES convergence rates for the solution of the NACA0012 case for DG discretization orders p = 1 to p = 4



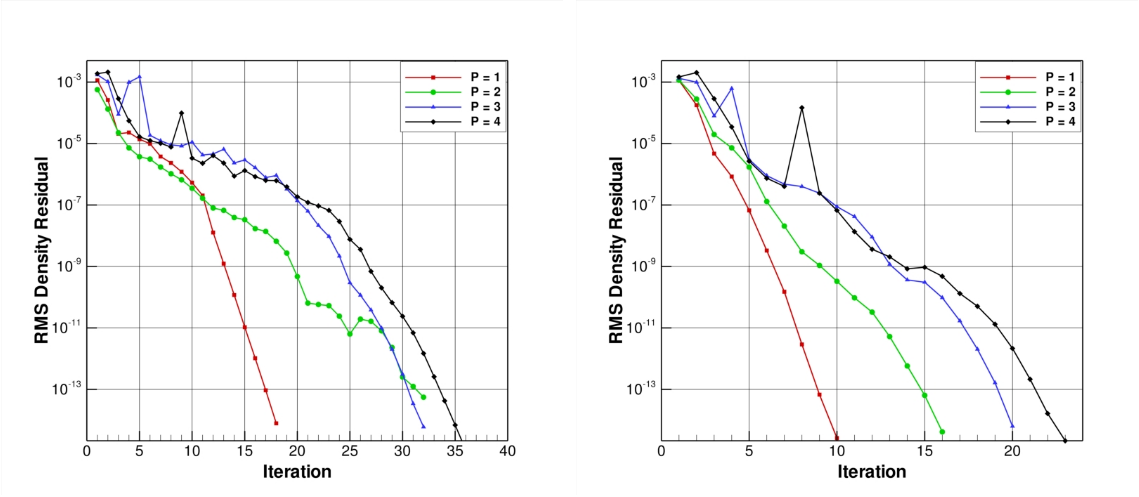
(a) Mesh with N = 7902 elements

(b) Surface curvature close-up

Figure 10. Two-element airfoil mixed element mesh used for solver comparison and local order-reduction robustness improvement

p = 2 solution is closer to that of the p = 1 solution. The overall p-independence is not improved very much for several reasons. The p = 3 and p = 4 solutions required the application of the local order-reduction technique, which combined with the fixed small number of linear iterations has caused a degradation of the p-independence compared to the NACA0012 case. Furthermore, each solver had a similar number of rises in the residual over their convergence history, which affects the MGPC-GMRES solver more severely because a single Newton iteration reprents double the percentage of the total number of Newton iterations compared to the linear multigrid solver. However, the NACA0012 results and the p = 2 result here show very good p-independence, leading us to conclude that the results for p = 3 and p = 4 cases are affected by the under-resolved stagnation point and or the application of local order-reduction without which the p = 3 and p = 4 solutions could not converge.

Though the MGPC-GRMES solver does not obtain textbook p-independence we prefer this solver to the linear



(a) Linear multigrid

(b) MGPC-GMRES

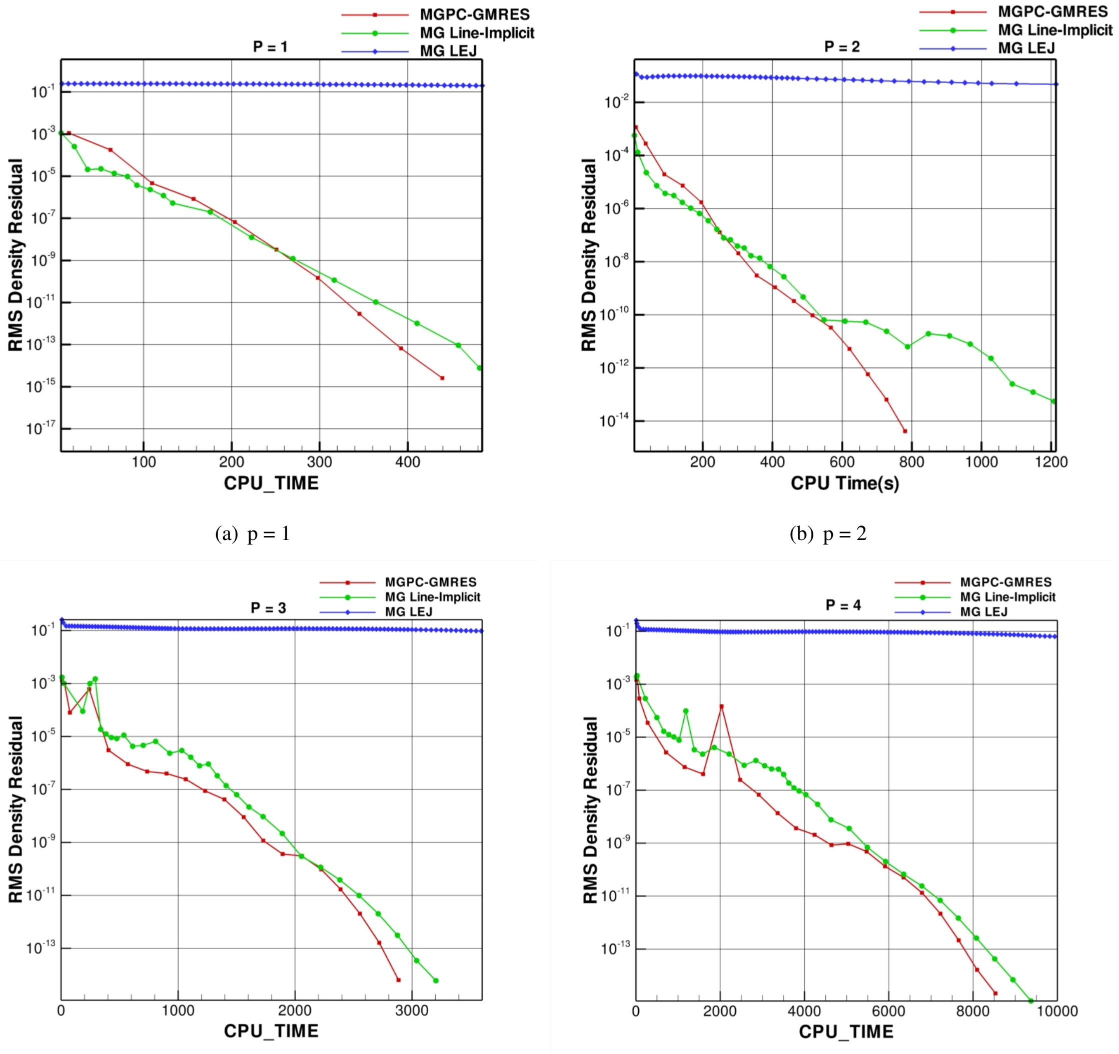
Figure 11. Convergence Rate of the two-element airfoil case for orders p = 1 to p = 4 using linear multigrid and MGPC-GMRES

multigrid one because it has proven to be faster in terms of overall CPU time. A CPU time comparison between the linear multigrid and MGPC-GMRES solvers is shown in Figures 12(a) through 12(d). These figures clearly demonstrate that the MGPC-GMRES solver is faster in terms of overall CPU time than the linear multigrid solver. On average the MGPC-GMRES solver is about 15% faster than the linear multigrid solver with the p = 2 case showing the most dramatic improvement. Figures 12(a) through 12(d) also illustrate the convergence obtained from these cases when the line-implicit Jacobi smoother is replaced by the LEJ smoother. Clearly not using the line-implicit smoother yields a very slow and inefficient solver for DG discretizations on anisotropic meshes.

VIII.C.2. Robustness Enhancement

This case was also used to test the local order-reduction technique for enhanced robustness and one should make note of the presense of several rises in the residual over the convergence history of this case. This particular mesh demonstates the under mesh-resolved phenomena discussion in Section VII.A. Near the stagnation point at the leading edge of main airfoil the mesh is too coarse to smoothly capture the high density and pressure gradients. As a result for p = 3 and p = 4 the extrema become non-smooth resulting in solver failure. To alleviate this problem we have applied the local order-reduction technique outlined in SectionVII.A. By applying this technique we have been able to generate solutions for p = 3 and p = 4 where we were previously unable to do so. Figures 13(a) and 13(b) show the cells and density contours where the order has been reduced for the p = 4 case. The “troubled” cell has vary large density variations across it and the density contours show some oscillations. One should make note of the rises in the residual convergence history for the p = 3 and p = 4 case. These rises simply correspond to the iterations where the order is being reduced. In these test cases density was the quantity used in the detector given by equation (22).

In conclusion the MGPC-GMRES solver represents a significant improvement over linear multigrid methods for DG discretizations due to the faster overall CPU time. Figures 14(a) and 14(b) show the Mach contours and stream lines for a p = 2 solution and Figures 15(a) and 15(b) show the same for a p = 4 solution. Note that the wake is very well preserved due to the anisotropic mesh and high-order accurate solutions. This case has also demonstrated the enhanced robustness given by the local order-reduction technique, without which the p = 3 and p = 4 solutions could not converge.



(a) p = 1

(b) p = 2

(c) p = 3

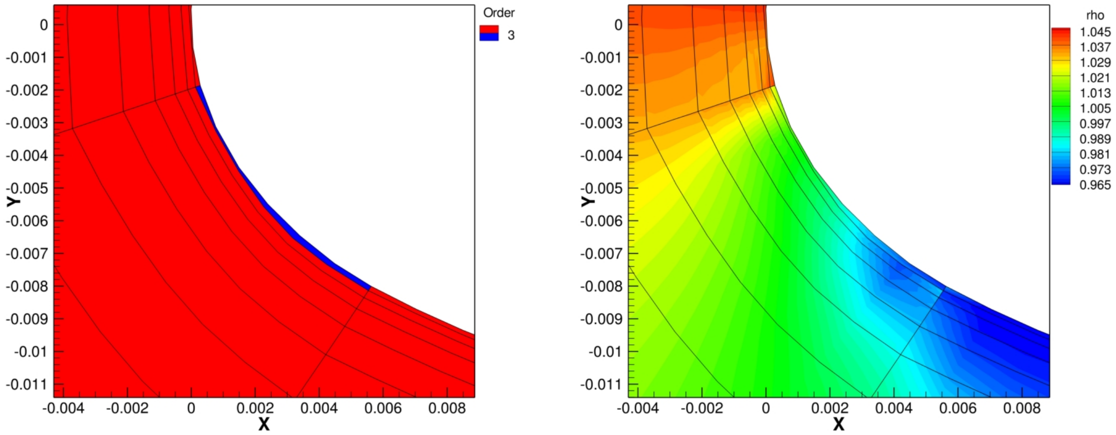
(d) p = 4

Figure 12. CPU time comparison between MGPC-GMRES (line-implicit Jacobi smoother), linear multigrid (line-implicit Jacobi smoother) and linear multigrid (LEJ smoother) for solution of the two-element airfoil case for using DG discretization orders p = 1 to p = 4

VIII.D. Turbulent Flat-Plate

Currently the first author is extending the above methods to the Reynolds Averaged Navier-Stokes (RANS) equations. In this case the RANS equations have been coupled to the one-equation turbulence model of Spalart and Allmaras(S.A. model). Here we show some preliminary results of that effort, specifically the incompressible zero pressure gradient flow over a flat plate. These results were generated with the form of the model found in reference,[43] that has been cast in conservative form.[44] This represents a verification of the RANS implementation for orders p = 1 to p = 4.

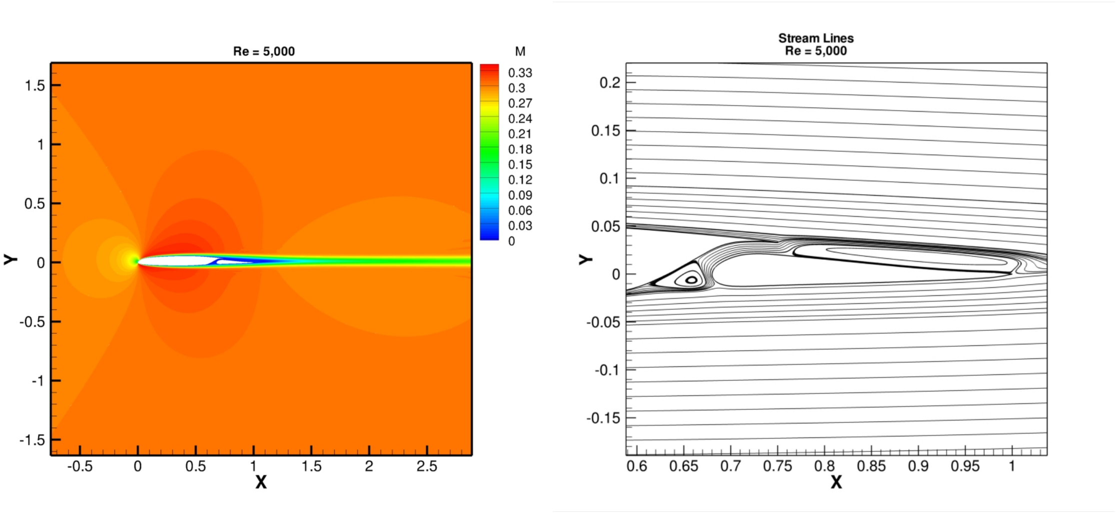
This case was run at the following conditions $M_\infty = .1$, $\alpha = 0^o$, and $Re = 10,000,000$. The computational mesh is made up of 540 quadrilateral elements and is shown in Figure 16. The velocity profiles for all orders are plotted along with experiment[45] in Figure 17(a). Additionally the skin friction is plotted in Figure 17(b). These results show good agreement with experiment for both velocity and skin friction, with the exception of the singularity at the leading edge of the plate where the mesh is too coarse. These results represent a preliminary attempt at solving the RANS equations. It has been shown in references[26,44] that the original form of the S.A. model yields undershoots at the boundary layer edge, which in turn generate negative values of the S.A. working variable. Figure 18 shows the turbulence variable



(a) Element order

(b) Density contours

Figure 13. Close-up of the two-element airfoil under mesh-resolved leading edge showing the reduced-order cell and density contours for p = 4 DG



(a) Mach contours

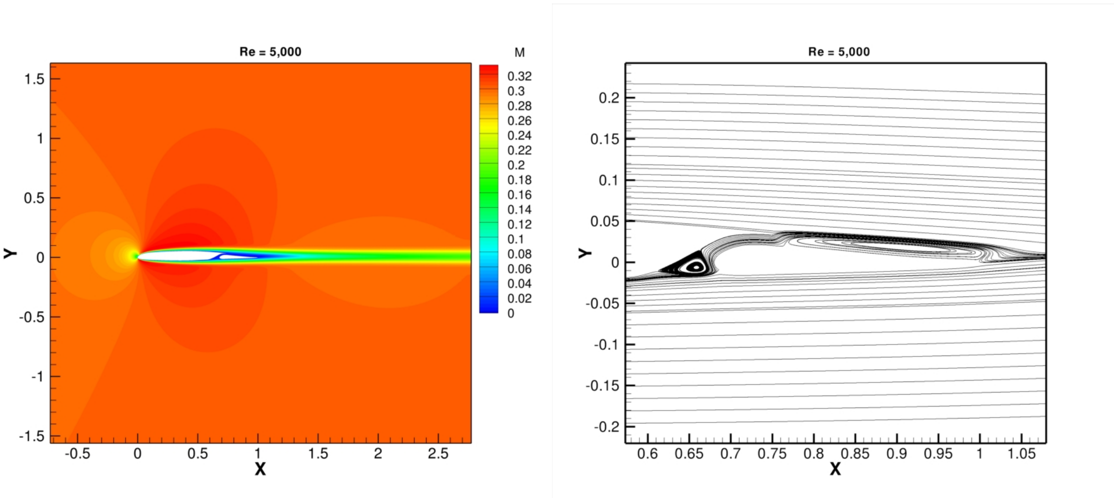
(b) Streamlines

Figure 14. Illustration of computed solution using DG for the laminar viscous flow over the two-element airfoil at $M_\infty = .3$, $\alpha = 0^o$, and $Re = 5,000$ for p = 2

plotted for the mid-chord of the plate for orders p = 1 and p = 4, noting the oscillations at the boundary layer edge, indicating the presence of non-smooth behavior. The convergence history using a p = 2 DG discretization is depicted in Figure 19. The MGPC-GMRES solver achieves convergence in under 80 Newton cycles for both the flow and turbulence model equations.

VIII.E. BGK Flux Results

Gas Kinetic(BGK) fluxes are very attractive for use with DG discretizations because they have the ability to compute both the inviscid and viscous inter-cell fluxes in a unified manner, without the need to resort to the use of a penalty or mixed method. Mixed methods (such as the one we have used thus far) have the disadvantage in that they require two additional terms on each face in the mesh for a total of three viscous terms, in contrast to only one term for the



(a) Mach contours

(b) Streamlines

Figure 15. Illustration of computed solution using DG for the laminar viscous flow over the two-element airfoil at $M_\infty = .3$, $\alpha = 0^o$, and $Re = 5,000$ for p = 4

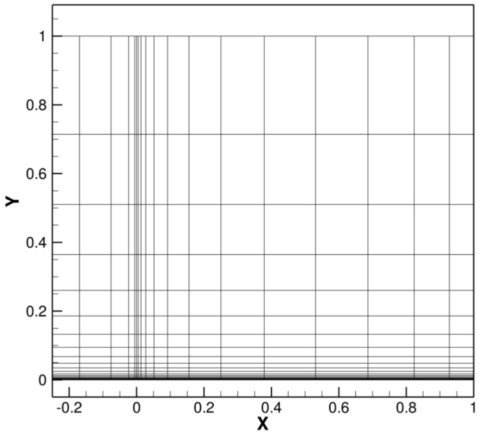
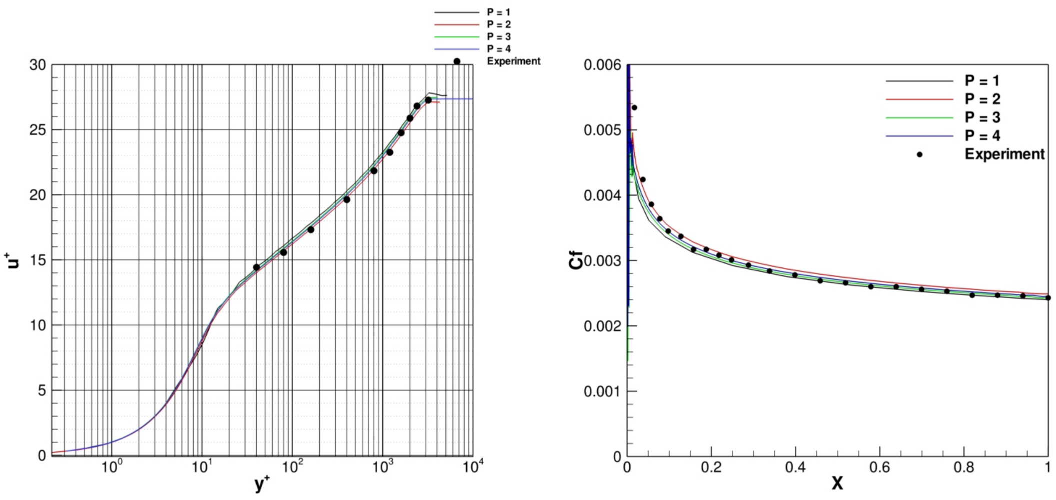


Figure 16. Mesh used for computing the turbulent flow over a flat plate with the Spalart-Allmaras turbulence model consisting of N = 540 Quadrilaterals

convective part. BGK methods have the ability to compute the viscous and inviscid fluxes simultaneously with only one flux evaluation. Furthermore, the SIP method (and similar methods) have rather ad hoc stabilization parameters associated with them. Though in this work we have removed some of the ambiguity associated with the SIP method, the expression for the penalty parameter cannot be proven to be sufficiently large for the Navier-Stokes equations (it can only be proven for Poisson's equation[16]). Additionally, kinetic-based fluxes have various other well-documented advantages in terms of accuracy and positivity[42]

Thus far, the BGK flux has been implemented for the 1D Euler equations and is used only in the $\mathbf{F}_\mathbf{c}^*(\mathbf{U}_p)\cdot\mathbf{n}$ term (i.e. the inter-element numerical flux) while the flux in the volume integral is computed as previously. Results for the periodic convection of an acoustic pulse using N = 10 elements are shown in Figures 20(a) through 20(e). The results show the pulse after four periods of convection. These results demonstrate that the convective form of the BGK flux results in a flux which is consistent with the volume flux term, implemented as previously, by integrating the analytical form of the flux in the continuous Euler equations.

Yet another advantage of the BGK flux is it's ability to compute flows where many traditional Riemann solvers fail



(a) u-velocity profile

(b) skin-friction coefficient

Figure 17. Comparison of computed solution using a DG discretization with the Spalart-Allmaras turbulence model for a flat plate boundary layer compared with experimental data from reference[45] for p = 1 up to p = 4

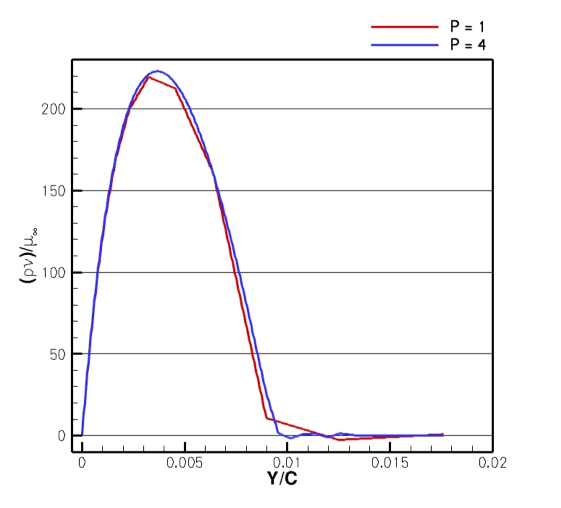


Figure 18. Computed profile of Spalart Allmaras working variable for turbulent flow over a flat plate at x/L = .5 (plate mid-chord) illustrating non-smooth behavior at the boundary layer edge for p = 1 and p = 4

or are overly dissipative, such as supersonic expansions. A comparison of a Mach 8 expansion computed via the BGK and Rusanov fluxes is shown in Figures 21(a) and 21(b) which demonstrate the ability of the BGK flux to compute this strong expansion. Clearly the local Rusanov flux can compute this expansion but is significantly more dissipative as shown by Figure 21(b). The BGK flux presents a very attractive alternative to traditionally used numerical Riemann solvers, and work is underway to extend this to one and two-dimensional viscous flows.

VIII.F. Three-Dimensional Backward Facing Step

The viscous fluxes together with the SIP method have been implemented in the previously developed three-dimensional inviscid flow code.[4] Current implementation is limited to tetrahedral meshes only. Preliminary 1-level results have been obtained for a backward-facing step configuration. Initial conditions are prescribed by Mach number, $M = 0.3$,

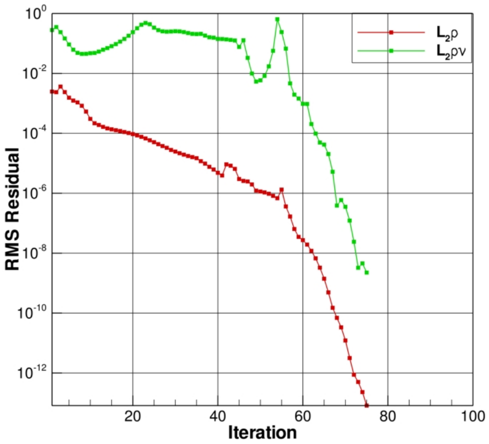


Figure 19. Convergence history of the density and turbulence model variable for the turbulent flow over a flat plate using the Spalart-Allmaras turbulence model for a p = 2 DG discretization using MGPC-GMRES solver

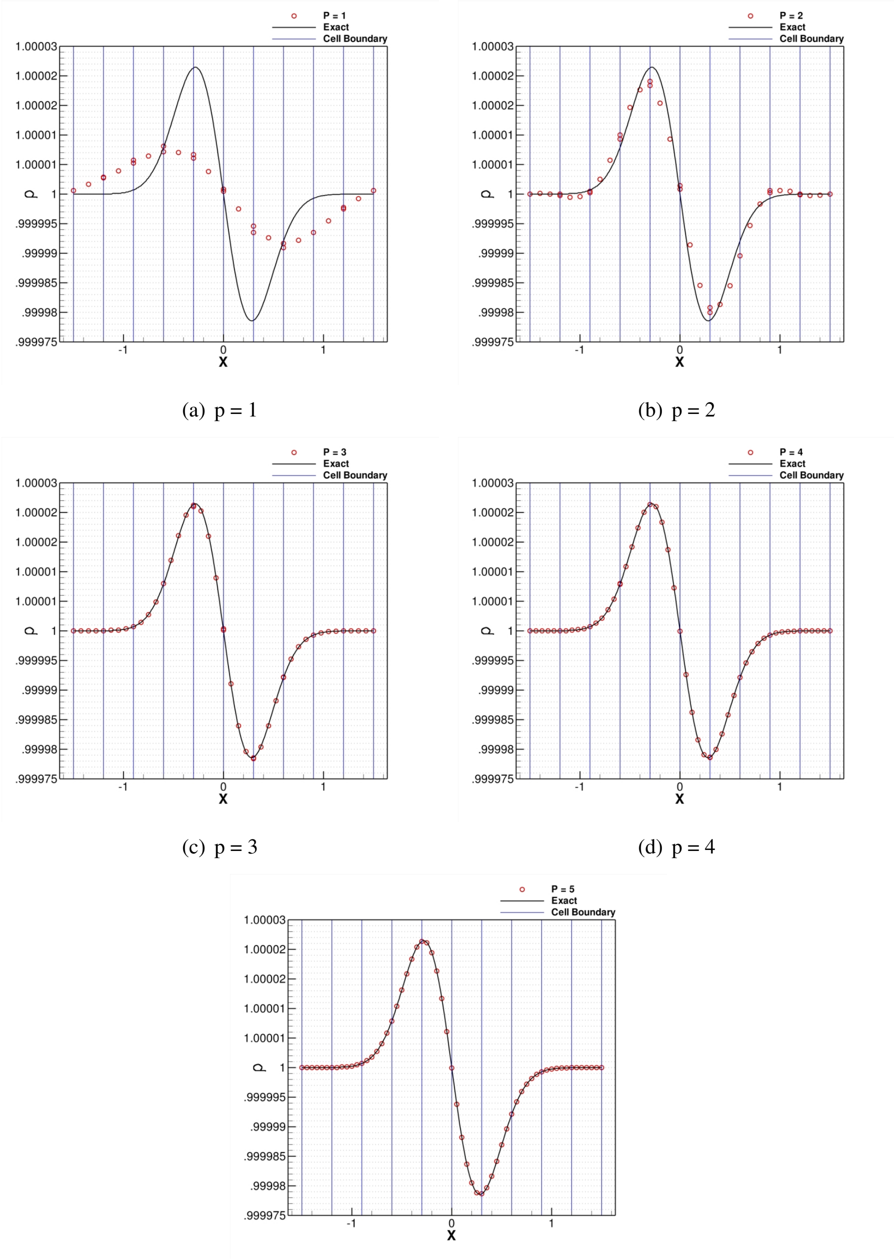
and Reynolds number, $Re = 500$. Figure 22(a) and 22(b) shows the geometry configuration and mesh discretization. Figure 22(c) depicts Mach iso-contours on one streamwise and two cross-stream stations together with the flow streamlines.

IX. Conclusions and Future Work

In this work, we have investigated and developed efficient solution strategies for steady-state viscous flows using high-order DG discretizations in the presence of curved, hybrid-element anisotropic meshes. The solution strategy is based on the *hp*-multigrid approach previously developed for inviscid flows. A line creation algorithm and line-implicit Jacobi smoother have been developed and implemented to enable efficient solution techniques on anisotropic meshes. Further improvement has been demonstrated through the use of a preconditioned Newton-Krylov technique. The methodology is developed for the two- and three-dimensional Navier-Stokes equations on unstructured anisotropic grids, using linear multigrid schemes.

Two-dimensional results are presented for a flat plate boundary layer, flow over a NACA0012 airfoil and a two-element airfoil. Current results demonstrate convergence rates which are nearly independent of the degree of anisotropy, order of accuracy (p) of the discretization and level of mesh resolution (h). It has been shown that the MGPC-GMRES algorithm out performs standard multigrid techniques both in terms of CPU time and optimality. MGPC-GMRES exhibits better h- and p-independence when the linear problem is fully converged as in the NACA0012 case. However, this might prove impractical for more realistic cases (*i.e* the two-element airfoil). A more practical scenario is to limit the number of multigrid cycles per Newton iteration. In this case MGPC-GMRES has been shown to be faster than linear multigrid in terms of CPU time for the same number of multigrid cycles. Preliminary results have demonstrated the ability of using these techniques for solving the Reynolds-averaged Navier-Stokes equations where the effect of turbulence is included through the fully coupled solution of an additional turbulence modeling equation. Additionally, the incorporation of kinetic-based flux functions has been investigated and shows promise for use with DG discretizations of the Navier-Stokes equations.

One of the principal issues with high-order discretizations is the robustness of these methods in dealing with discontinuous, non-smooth, or under-resolved features. In this work, we have investigated an element-wise order-reduction technique for addressing such robustness issues. By detecting and reducing the order of "troubled" cells for



(a) p = 1

(b) p = 2

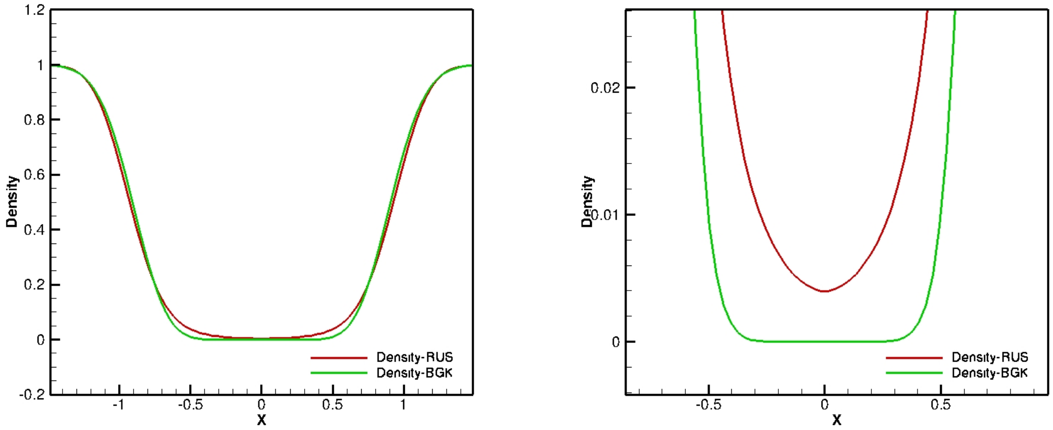
(c) p = 3

(d) p = 4

(e) p = 5

Figure 20. Acoustic pulse after four periods of convection using the BGK flux on N = 10 elements for p = 1 through p = 4

the two-element airfoil problem we have been able to overcome the under resolved stagnation point. For this case the issue stems from an under mesh-resolved leading edge where gradients in flow quantities are high. By under resolving the geometry here what should be smooth extrema become non-smooth resulting in the solver creating overshoots that then cause failure. By locally reducing the order of accuracy, the increased dissipation allows the solver to handle the overshoot and converge with this under-resolved region. Although in this work we have implemented a local order-reduction technique, raising the order high enough should also help the problem because with enough resolution the extrema should again become smooth. An alternate technique for robustness improvement would be the addition of



(a) Density

(b) Close-up of near vacuum region

Figure 21. Mach 8 expansion test case using p = 0 and N = 100 elements, computed with BGK and Rusanov fluxes

explicit artificial dissipation such as in reference.[40] This approach is attractive because it does not reduce the order of the solution in the cell thus potentially limiting the overall degradation of accuracy.

Future work will concentrate on investigating additional techniques for handling under-resolved or non-smooth phenomena robustly and accurately, while also extending these techniques to the three-dimensional Navier-Stokes equations using hybrid-element topologies, for complex flow problems.

X. Acknowledgments

The first author would like to thank Dr.'s Li Wang(Univ, of Tenn. Chatanooga) and Khosro Shahbazi(Brown Univ.) and Mr. Brian Lockwood(Univ. of Wyoming) for their many helpful discussions in preparing this work. This work was made possible thanks to funding from NASA under grant number NNX07AC31A and from support under a Phase 1 STTR project supported by AFOSR contract F9550-09-C-0021.

References

[1] Laflin, K., Brodersen, O., Rakowitz, M., Vassberg, J., Wahls, R., and Morrison, J., "Summary of Data from the Second AIAA CFD Drag Prediction Workshop," AIAA Paper 2004-0555.

[2] Vassberg, J. C., Tinoco, E. N., Mani, M., Brodersen, O. P., Eisfeld, B., Wahls, R. A., Morrison, J. H., Zickuhr, T., Laflin, K. R., and Mavriplis, D. J., "Summary of the Third AIAA CFD Drag Prediction Workshop," AIAA Paper 2007-0260.

[3] Nastase, C. R. and Mavriplis, D. J., "High-Order discontinuous Galerkin Methods using an hp-muligrid approach," *J. Comput. Phys.*, Vol. 213, No. 1, mar 2006, pp. 330–357.

[4] Nastase, C. R. and Mavriplis, D. J., "A Parallel hp-Multigrid Solver for Three-Dimensional Discontinuous Galerkin Discretizations of the Euler Equations," *Proceedings of 45th Aerospace Sciences Meeting and Exihibit, Reno NV*, 2007, AIAA Paper 2007–0512.

[5] Wang, L. and Mavriplis, D. J., "Implicit solution of the unsteady Euler equations for high-order accurate discontinuous Galerkin discretizations," *J. Comput. Phys.*, Vol. 225, No. 2, Aug 2007, pp. 1994–2015.

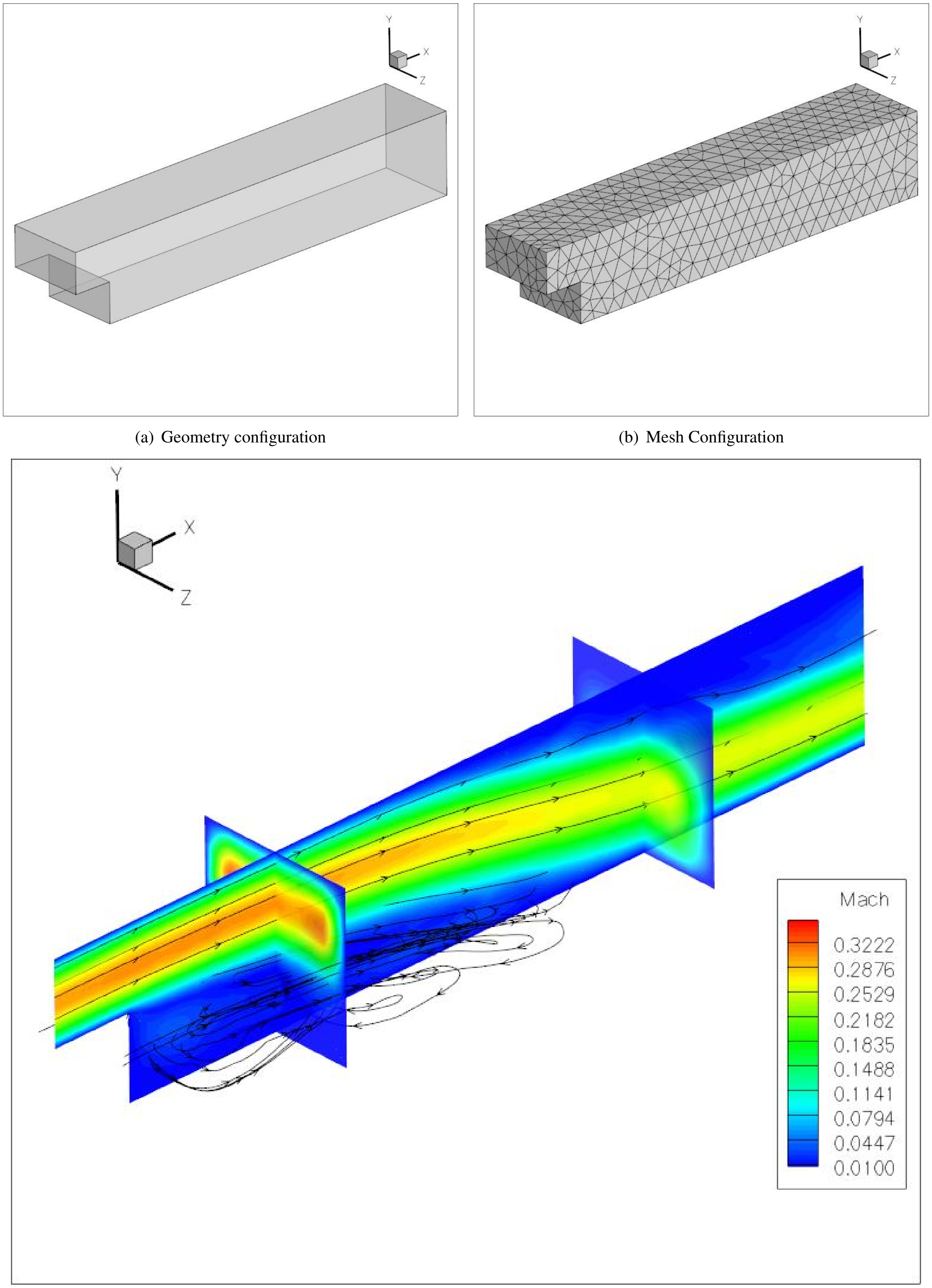
[6] Bassi, F. and Rebay, S., "High-Order Accurate Discontinuous Finite Element Solution of the 2D Euler Equations," *J. Comput. Phys.*, Vol. 138, 1997, pp. 251–285.

[7] Cockburn, B. and Shu, C.-W., "The Local Discontinuous Galerkin Method for Time-Dependent Convection-Diffusion Systems," *SIAM J. Numer. Anal.*, Vol. 35, No. 6, 1998, pp. 2440–2463.

[8] Warburton, T. C., Lomtev, I., Du, Y., Sherwin, S. J., and Karniadakis, G. E., "Galerkin and Discontinuous Galerkin Spectral/*hp* Methods," *Comput. Methods Appl. Mech. Engrg.*, Vol. 175, 1999, pp. 343–359.

[9] Cockburn, B. and Shu, C.-W., "Runge-Kutta Discontinuous Galerkin Methods for Convection-Dominated Problems," *SIAM J. Sci. Comput.*, Vol. 16, No. 3, 2001, pp. 173–261.

[10] Davis, S. F., "Simplified Second-Order Godunov-Type Methods," *SIAM J. Sci. Statist. Comput.*, Vol. 9, No. 3, 1988, pp. 445–473.



(a) Geometry configuration

(b) Mesh Configuration

(c) Mach number

Figure 22. Three dimensional DG solution for flow over a backward facing step at $M = .3$ $Re = 500$ **for p = 2**

[11] Roe, P. L., “Approximate Riemann Solvers, Parameter vectors, and Difference Schemes,” *J. Comput. Phys.*, Vol. 43, 1981, pp. 357–372.

[12] Harten, A., Lax, P. D., and Van Leer, B., “On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws,” *SIAM Review*, Vol. 25, No. 1, 1983, pp. 35–61.

[13] Toro, F. E., *Riemann Solvers and Numerical Methods for Fluid Dynamics*, Applied Mechanics, Springer-Verlag, New York, NY, 1999.

[14] Batten, P., Clarke, N., Lambert, C., and Causon, D. M., “On the Choice of Wavespeeds for the HLLC Riemann Solver,” *SIAM J. Sci. Comput.*, Vol. 18, No. 2, 1997, pp. 1553–1570.

[15] Batten, P., Leschiner, M. A., and Goldberg, U. C., “Average-State Jacobians and Implicit Methods for Compressible Viscous and Turbulent Flows,” *J. Comput. Phys.*, Vol. 137, 1997, pp. 38–78.

[16] Shahbazi, K., *A Parallel High-Order Discontinuous Galerkin Solver For The Unsteady Incompressible Navier-Stokes Equations in Complex Geometries*, Ph.D. thesis, University of Toronto, may 2007.

[17] Hartmann, R. and Houston, P., “Symmetric Interior Penalty DG Methods for the Compressible Navier-Stokes Equations I: Method Formulation,” *Internal Journal of Numerical Analysis and Modeling*, Vol. 3, No. 1, 2006, pp. 1–20.

[18] Shahbazi, K., Mavriplis, D. J., and Burgess, N. K., “Multigrid Algorithms for High-Order Discontinuous Galerkin Discretizations of the Compressible Navier-Stokes Equations,” *J. Comput. Phys.*, Vol. 228, No. 21, Nov 2009, pp. 7917–7940.

[19] Solin, P., Segeth, P., and Zel, I., *High-Order Finite Element Methods*, Studies in Advanced Mathematics, Chapman and Hall, 2003.

[20] Szabo, B. and Babuska, I., *Finite Element Analysis*, John Wiley & Sons, Inc., New York, NY, 1991.

[21] Dunavant, D. A., “Highr Degree Efficient Symmetrical Gaussian Quadrature Rules for the Triangle,” *Int. J. Numer. Meth. Engng.*, Vol. 21, 1985, pp. 1129–1148.

[22] Dunavant, D. A., “Economical Symmetrical Quadrature Rules for Complete Polynomials Over a Square Domain,” *Int. J. Numer. Meth. Engng.*, Vol. 21, 1985, pp. 1777–1784.

[23] Cockburn, B., Hou, S., and Shu, C.-W., “The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case,” *Math. Comput.*, Vol. 54, No. 545, 1990.

[24] Mavriplis, D. J., “Revisiting the Least-Squares Proceedure for Gradient Reconstruction on Unstrucutred Meshes,” *Proceeding of the 16th Computational Fluid Dynamics Conference, Orlando, FL*, June 2003, AIAA Paper 2003-3986.

[25] Persson, P.-O. and Peraire, J., “Curved Mesh Generation and Mesh Refinement using Lagrangian Solid Mechanics,” *Proceeding of the 47th Aerospace Sciences Meeting, Orlando, FL*, Jan. 2009, AIAA Paper 2009-949.

[26] Oliver, T. A., *A high-order, adaptive, discontinuous Galerkin finit element method for the Reynolds averaged Navier-Stokes equations*, Ph.D. thesis, Massachusetts Institute of Technology, sept 2008.

[27] Trottenberg, U., Schuller, A., and Oosterlee, C., *Multigrid*, Academic Press, London, UK, 2000.

[28] Mavriplis, D. J., “An Assessment of Linear versus Non-Linear Multigrid Methods for Unstructured Mesh Solvers,” *J. Comput. Phys.*, Vol. 175, 2002, pp. 302–325.

[29] Nastase, C. R. and Mavriplis, D. J., “High-Order Discontinuous Galerkin Methods using a Spectral Multigrid Approach,” *Proceedings of the 43rd Aerospace Sciences Meeting and Exihibit, Reno NV*, 2005, AIAA Paper 2005–1268.

[30] Fidkowski, K. J., Oliver, T. A., Lu, J., and Darmofal, D. L., “p-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier-Stokes equations,” *J. Comput. Phys.*, Vol. 207, No. 1, Feb 2005, pp. 92–113.

[31] Fidkowski, K. J. and Darmofal, D. L., “Development of a Higher-Order Solver for Aerodynamic Applications,” *Proceedings of the 42nd Aerospace Sciences Meeting and Exihibit, Reno NV*, 2004, AIAA Paper 2004–0436.

[32] Mavriplis, D. J., “Multigrid Techniques for Unstructured Meshes,” *VKI Lecture Series VKI-LS 1995-02*, mar 1995.

[33] Trottenberg, U., Oosterlee, C., and Schuller, A., *Multigrid*, Elsevier Academic Press, San Diego, CA, 2001.

[34] Mavriplis, D. J., “Directional Agglomeration Multigrid Techniques for High-Reynolds Number Viscous Flow Solvers,” *AIAA Journal*, Vol. 37, No. 10, Oct. 1999, pp. 1222–1230.

[35] Thomas, J. W., *Numerical Partial Differential Equations: Finite Difference Methods*, Texts in Applied Mathematics, Springer, New York, NY, 1995.

[36] Trefethen, L. N. and Bau, D. I., *Numerical Linear Algebra*, SIAM, Philadelphia, PA, 1997.

[37] Greenbaum, A., *Iterative Methods for Solving Linear Systems*, Frontiers in Applied Mathematics, SIAM, Philadelphia, PA, 1997.

[38] Saad, Y., *Iterative Methods for Sparce Linear Systems*, SIAM, Philadelphia, PA, 2003.

[39] Golub, G. H. and Van Loan, C. F., *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1996.

[40] Persson, P.-O. and Peraire, J., “Sub-Cell Shock Capturing for Discontinuous Galerkind Methods,” *Proceedings of 44th Aerospace Sciences Meeting and Exihibit, Reno NV*, Jan. 2006, AIAA Paper 2006-112.

[41] Barter, G. E. and Darmofal, D. L., “Shock Capturing with Higher-Order, PDE-based Artificial Viscosity,” *Proceeding of the 18th Computational Fluid Dynamics Conference, Miami FL*, June 2007, AIAA Paper 2007-3823.

[42] Kim, C. A., Jameson, A., Martinelli, L., and Xu, K., “An Accurate LED-BGK Solver on Unstructured Adpative Meshes,” *Proceeding of the 35th Aerospace Sciences Meeting, Reno, NV*, Jan. 1997, AIAA Paper 1997-238.

[43] Spalart, P. and Allmaras, S., “A one-equation turbulence model for Aerodynamic flows,” *Le Recherche Aerospátiale*, Vol. 1, 1994, pp. 5–21.

[44] Oliver, T. A. and Darmofal, D. L., “Impact of Turbulence Model Irregularity on High-Order Discretizations,” *Proceeding of the 47th Aerospace Sciences Meeting and Exhibit, Orlando FL*, Jan. 2009, AIAA Paper 2009-953.

[45] Coles, D. and Hirst, E., “Computation of Turbulent Boundary Layers,” *AFOSR-IFP-Stanford Conference*, 1969, Vol. II Stanford University.