Aachen Institute for Advanced Study in Computational Engineering Science

Preprint: AICES-2008-12

31/December/2008

Efficient Algorithms for High-Order Discretizations of the Euler and Navier-Stokes Equations

G. May, F. Iacono and A. Jameson

RWTH AACHEN UNIVERSITY

AICES

Financial support from the Deutsche Forschungsgemeinschaft (German Research Association) through grant GSC 111 is gratefully acknowledged.

List of AICES technical reports: http://www.aices.rwth-aachen.de/preprints

Efficient Algorithms for High-Order Discretizations of the Euler and Navier-Stokes Equations

Georg May* and Francesca Iacono†

AICES Graduate School, RWTH Aachen, Schinkelstr. 2, 52056 Aachen, Germany

Antony Jameson‡

Department of Aeronautics and Astronautics, Stanford University, Durand Building, Stanford, CA 94305

Higher order discretizations have not been widely successful in industrial applications to compressible flow simulation. Among several reasons for this, one may identify the lack of tailor-suited, best-practice relaxation techniques that compare favorably to highly tuned lower order methods, such as finite-volume schemes. In this paper we investigate efficient Spectral Difference discretization methods for the Euler and Navier-Stokes equations, and present solution algorithms using such techniques as h/p-Multigrid. We present a novel hybrid multilevel relaxation method that combines matrix-free implicit relaxation techniques with explicit time-stepping using geometric multigrid.

I. Introduction

Many have anticipated the arrival of high-order discretization methods for compressible fluid flow in complex domains as the CFD method of choice. However, for industrial applications in external aerodynamics lower order methods, such as finite-volume schemes, are still by far more popular. The reasons for this are several. Firstly, shock capturing poses a much bigger challenge for higher order discretization methods, compared to high-resolution finite volume schemes. Often one encounters severe convergence degradation when heavy use of limiters is required. Such limiters typically alter the frequency spectrum with very heavy damping of higher order modes, and careful limiting of lower order modes, which is a much more intrusive procedure if the local spectrum is richer to begin with. Secondly, for established CFD methodologies tailor-suited convergence acceleration techniques have emerged over the past decades. High-order solvers thus compete with very mature technology, and novel discretization techniques have to be augmented with extremely efficient solutions algorithms.

Numerical schemes of third or higher spatial order are not yet efficient enough for many problems of practical engineering interest, in particular high-speed viscous flow. For steady problems a fast solution technique is necessary to solve the arising systems of nonlinear equations. This is the focus of the present paper. We investigate efficient solution techniques, such as h/p-multigrid methods and (matrix-free) implicit relaxation schemes for steady compressible flow. Trade-offs between computational cost, memory requirements, and also ease of implementation are considered.

Spatial discretization is implemented using the Spectral Difference scheme, which has been proposed as a collocation-based method with the aim to achieve efficiency by avoiding volume and surface quadratures, while maintaining conservation.[1–3] The Spectral Difference approach extends tensor-product-based collocation approaches that had previously been formulated for quadrilateral meshes[4] to more general unstructured-grid elements. The scheme may also be viewed as a particularly efficient nodal Discontinuous Galerkin scheme.[5] In the present paper we consider the Spectral Difference scheme for two-dimensional compressible fluid flow on triangular unstructured grids.

Focus is also on efficient implementation of solution algorithms. For nonlinear equations it is not always a priori clear which elements of the solution methodology, comprised of spatial discretization and relaxation

*Junior Research Group Leader, AIAA Member

†PhD candidate, AIAA student Member

‡Thomas V. Jones Professor of Engineering, AIAA Member

algorithms works best for a given problem. Thus one desires a flexible implementation that lets the user combine different building blocks, such as numerical flux functions, approximations to Jacobian matrices, and pre-conditioning, without extensive re-derivation or re-implementation of code. We use such tools as efficient numerical libraries and automatic differentiation (AD) to obtain a flexible platform designed to aid in the investigation into efficient solution methods for high-order discretizations of the Euler and Navier-Stokes equations.

II. Governing Equations

For inviscid compressible flow we solve the Euler equations in conservation form:

$$\frac{\partial w}{\partial t} + \sum_{j=1}^{d} \frac{\partial f_j}{\partial x_j} = 0 \ . \tag{1}$$

We consider the two-dimensional equations, i.e. $d = 2$. The vector of conservative variables w and the flux vectors f_j are given by

$$w = \begin{pmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ E \end{pmatrix} , \qquad f_j = \begin{pmatrix} \rho u_j \\ \rho u_j u_1 + p\delta_{j1} \\ \rho u_j u_2 + p\delta_{j2} \\ \rho u_j H \end{pmatrix} , \qquad j = 1, 2 \ . \tag{2}$$

Here ρ is the density, p is the pressure, E is the energy, and $H = (E + p)/\rho$ is the enthalpy. The velocity vector is given by $u = (u_1, u_2)^T$. We shall assume a thermally and calorically perfect gas, and hence close the equations by the equation of state

$$p = (\gamma - 1)\left(E - \frac{1}{2}\rho||u||^2\right) \ , \tag{3}$$

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

In the case of viscous flow the compressible Navier-Stokes equations in conservation form are solved:

$$\frac{\partial w}{\partial t} + \sum_{j=1}^{d} \frac{\partial (f_j - f_j^{(v)})}{\partial x_j} = 0 \ , \tag{4}$$

where for two-dimensional flow the viscous flux vectors are given by

$$f_j^{(v)} = \begin{pmatrix} 0 \\ \tau_{1j} \\ \tau_{2j} \\ \tau_{j1}u_1 + \tau_{j2}u_2 - q_j \end{pmatrix} , \qquad j = 1, 2 \ . \tag{5}$$

The stress tensor τ and heat flux vector q are written for a Newtonian Fluid under the Stokes hypothesis, and using the Fourier law of heat conduction

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} + \delta_{ij}\nabla \cdot u\right) \ , \tag{6}$$

$$q_j = -\frac{\mu}{Pr}\frac{\gamma}{\gamma - 1}\frac{\partial (p/\rho)}{\partial x_j} \ , \tag{7}$$

where μ is the dynamic viscosity. The Prandtl number $Pr = c_p\mu/\kappa$, can be assumed constant, with $Pr \approx 0.72$ for air. The specific heat at constant pressure, which is assumed to be constant, is denoted by c_p, while κ is the heat conduction coefficient. For the viscosity we use Sutherland's law

$$\frac{\mu}{\mu_0} = \left(\frac{T}{T_0}\right)^{\frac{3}{2}} \frac{T_0 + A}{T + A} \ , \tag{8}$$

where the constant A is taken as $A = 100.3$K, and T_0 is a reference temperature corresponding to the viscosity μ_0.

III. Discretization

III.A. Discretization for inviscid Flow

For moderate levels of accuracy, higher order numerical schemes are advantageous only if a particular discretization method supports efficient implementation. Local numerical integration required by standard Discontinuous Galerkin (DG) methods can be considered a drawback in this regard. For nonlinear conservation laws, where integration necessitates explicit evaluation of analytical and numerical flux functions at quadrature points, numerical quadratures are quite irksome, since a generic optimal node placement for a given accuracy is not known for general mesh elements, in particular simplex elements. In this context the Spectral Difference scheme has been proposed as a quadrature-free scheme based on local interpolation of the strong form of the governing equations.[1–3,6,7] Alternatively the scheme may be viewed as a particular nodal DG scheme based on the weak form of the equations that avoids both volume and surface quadratures by a suitable projection of flux functions and choice of local basis functions.[5]

Consider the conservation law (1) for $d = 2$ on some domain $\Omega \subset \mathbb{R}^2$, subject to suitable initial and boundary conditions. Consider a triangulation $\mathcal{T}_h = \{T_i\}$, such that $\overline{\Omega}_h = \bigcup \overline{T_i}$. Assume that there exist mappings $\Phi_i : \hat{T} \to T_i$ with nonsingular Jacobian $J_i = \partial x/\partial \xi$, such that each element in the triangulation can be mapped to a reference domain $\hat{T}$. For simplicity we consider straight-sided triangular elements in this section, although such a restriction is not necessary. (The implementation used in our solver allows for elements with curved edges.) Consider the finite-dimensional space $\mathcal{V}_h^m = \left\{ v \in L^2(\Omega_h) : v|_{T_i} \circ \Phi_i \in \mathcal{P}^m(\hat{T}) \right\}$, where $\mathcal{P}^m(\hat{T}) = \text{span}\left\{ \xi^\alpha : \xi \in \hat{T}\,, \alpha_i \geq 0\,, |\alpha| \leq m \right\}$, and $\alpha \in \mathbb{N}^2$ is a multiindex. The dimension of the space $\mathcal{P}^m$ is given by

$$N_m^{d=2} = \frac{(m+1)(m+2)}{2}\,. \tag{9}$$

As a particular basis for $\mathcal{P}^m(\hat{T})$ consider the fundamental polynomials L_j of multivariate Lagrangian interpolation, corresponding to a nodal set $\mathcal{S}_m = \{\xi_j\,, j = 1, \ldots, N_m^d\}$, i.e.

$$w_h|_{T_i} := \sum_{j=1}^{N_m} w_{i,j} L_j(\xi)\,, \tag{10}$$

where $w_j = w(\xi_j)$, and the representation (10) is understood to hold componentwise. This leads to the DG discretization

$$\sum_{j=1}^{N_m} \dot{w}_j \int_{\hat{T}} L_j L_k \, d\xi - \int_{\hat{T}} \nabla^\xi L_k \cdot (J^{-1} f(w_h)) \, d\xi + \int_{\partial \hat{T}} \widetilde{h}\, L_k \, ds = 0\,, \qquad k = 1, \ldots, N_m\,, \tag{11}$$

where the numerical flux function $\widetilde{h}$, consistent at any $\xi \in \partial\hat{T}$ with the normal flux $(J^{-1}f) \cdot n^\xi = f \cdot n$, has been introduced ($n^\xi$ and n are the outward pointing normals on $\partial\hat{T}$ and ∂T, respectively).

In the multidimensional case the quadrature-free DG approximation necessitates a projection of both the flux function and the numerical flux onto a finite-dimensional space using polynomials of maximum total degree $m+1$. Here we use multivariate interpolation on a nodal set $\mathcal{Q}_{m+1} = \{\hat{\xi}_j\,, j = 1, \ldots, N_{m+1}\}$ with $\hat{L}_j$ the corresponding fundamental polynomials. This leads to

$$\widetilde{f}_h := \sum_{j=1}^{N_{m+1}} \widetilde{f}_j \hat{L}_j(\xi)\,, \tag{12}$$

where the degrees of freedom are computed as

$$\widetilde{f}_k = \begin{cases} J^{-1} f(u_{i,k}) & , \quad \hat{\xi}_k \in \hat{T} \\ \widetilde{f}^{num} & , \quad \hat{\xi}_k \in \partial\hat{T} \end{cases}\,. \tag{13}$$

The coefficients $\widetilde{f}^{num}$ are chosen such that $\widetilde{f}^{num} \cdot n^\xi = \widetilde{h}$. Introduce the following assumption

Assumption III.1 *The restriction of the 2-dimensional nodal set $\mathcal{Q}_{m+1}$ to each face $e \in \partial\hat{T}$ supports a unique one-dimensional interpolation of order $m+1$ with corresponding Lagrangian fundamental polynomials $l_k(\zeta)$, where $\zeta \in \partial\hat{T} \subset \mathbb{R}$. We assume that a subset of exactly $N_{m+1}^1 = m+1$ points is located on each $e \in \partial\hat{T}$.*

Examples of nodes satisfy Assumption III.1 are abundant for the triangle.[8–11] It can be shown that under this assumption, if the numerical flux function h is interpolated using the restriction of the nodal set $\mathcal{Q}_{m+1}$, and using (13) for the collocation of f, the surface integral in eq. (11) vanishes.[5] Furthermore, well known identities for the Lagrangian basis functions,[5,12] lead to a scheme for each cell

$$\frac{du_{i,j}}{dt} + \sum_{k=1}^{N_{m+1}} (\nabla^{\xi} \hat{L}_k)\Big|_{\xi_j} \cdot \widetilde{f}_k = 0 \; . \tag{14}$$

This scheme is known as the Spectral Difference Scheme. In our implementation the numerical flux function h is given by Jameson's CUSP flux.[13]

III.B. Spectral Difference Discretization of Advection-Diffusion Systems

For brevity we consider the one-dimensional case, while the multidimensional formulation follows along the lines presented in section III.A. Consider the one-dimensional equation

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2} \; . \tag{15}$$

One way of solving Eq. (15) is to rewrite it as a system:

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = \nu \frac{\partial \sigma}{\partial x} \tag{16}$$

$$\frac{\partial u}{\partial x} - \sigma = 0 \; , \tag{17}$$

where formally $\sigma = \partial_x u$. (Note that we restrict this discussion to $\nu = const.$. For the extension to $\nu \neq const.$ we refer to Cockburn and Shu.[14]) This general approach has been taken by many researchers for viscous discretization in a Discontinuous Galerkin context,[14–16] and has since been formalized in a fairly generic way for elliptic equations.[17] For advection-diffusion systems and the Navier-Stokes equations we extend the inviscid Spectral Difference discretization to include a nodal form of these classical discretization methods for elliptic equations.[17] As in the purely hyperbolic case, one may integrate against smooth test functions w, v, to obtain the weak formulation,

$$(u_t, v) - (f - \nu\sigma, v') + < f - \nu\sigma, v > \; = \; 0 \; , \tag{18}$$

$$(\sigma, w) + (u, w') - < u, w > \; = \; 0 \; , \tag{19}$$

where for brevity we use $(\cdot,\cdot)$ and $<\cdot,\cdot>$ to denote volume integral and surface terms, respectively. Consider a partition of the real line into intervals $I_i = (x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}})$ with $\Delta x_i = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}$, and define a transformation to a reference domain via linear mapping $\Phi_i : [-1,1] \to I_i$. Consider the finite-dimensional space $\mathcal{V}_h^m$ of bounded functions v for which $v|_{I_i} \in \mathcal{P}^m$, where $\mathcal{P}^m$ is the space of polynomials of degree at most m. Consider nodal sets $\mathcal{S}_m = \{\xi_k; k = 0, \ldots, m\}$ and $\mathcal{Q}_{m+1} = \{\hat{\xi}_k; k = 0, \ldots, m+1\}$, where for the latter we impose the restriction that the end points $|\xi| = 1$ be included in the set, and we may suppose that $\hat{\xi}_0 = -1$, and $\hat{\xi}_m = 1$. The corresponding Lagrangian fundamental polynomials $l_j(\xi_k) = \delta_{jk}$, and $\bar{l}_j(\hat{\xi}_k) = \delta_{jk}$ form bases for $\mathcal{V}_h^m$, and $\mathcal{V}_h^{m+1}$, respectively. The Discontinuous Galerkin approximation reads for each element i: find $u_h \in \mathcal{V}_h^m$ and $\sigma_h \in \mathcal{V}_h^{m+1}$ such that

$$\int_{-1}^{1} (\partial_t u_h)\, l_j \, d\xi - \frac{2}{\Delta x_i} \left\{ \int_{-1}^{1} g_h l_j' \, d\xi + \hat{g}_{i+\frac{1}{2}} l_j(1) \delta_{j,m+1} - \hat{g}_{i-\frac{1}{2}} l_j(-1) \delta_{j0} \right\} \; = \; 0 \; , \quad j = 0, \ldots m \; , \tag{20}$$

$$\int_{-1}^{1} \sigma_h \, \bar{l}_j \, d\xi + \frac{2}{\Delta x_i} \left\{ \int_{-1}^{1} u_h \, \bar{l}_j' \, d\xi - (\hat{u}_{i+\frac{1}{2}} \bar{l}_j(1) \delta_{i,m+1} - \hat{u}_{i-\frac{1}{2}} \bar{l}_j(-1) \delta_{i,0}) \right\} \; = \; 0 \; , \quad j = 0, \ldots m+1 \; , \tag{21}$$

where $g = f - \nu\sigma$ has been defined. There appear three different numerical fluxes, $\hat{f}$, $\hat{\sigma}$, and $\hat{u}$. In line with the quadrature-free approach used in the purely hyperbolic case, define suitable projections for the terms in the volume integrals. The approximation for the flux is written:

$$f_h|_{I_i} = \sum_{j=0}^{m+1} \bar{l}_j \widetilde{f}_j \; , \qquad \widetilde{f}_j = \begin{cases} \hat{f}_{i-\frac{1}{2}} & , \quad j = 0 \; , \\ \hat{f}_{i+\frac{1}{2}} & , \quad j = m+1 \; , \\ f(u_h(x_j)) & , \quad \text{otherwise} \; . \end{cases} \tag{22}$$

Although the remaining terms in the volume integrals in (20) and(21), i.e. u_h and σ_h are linear in the basis functions, and hence support a quadrature-free formulation without modification, we replace u_h and σ_h with a projection that also uses the numerical fluxes at the cell boundaries:

$$\widetilde{\sigma}_h|_{I_i} = \sum_{j=0}^{m+1} \bar{l}_j \widetilde{q}_j \ , \qquad \widetilde{\sigma}_j = \begin{cases} \hat{\sigma}_{i-\frac{1}{2}} & , \quad j = 0 \ , \\ \hat{\sigma}_{i+\frac{1}{2}} & , \quad j = m+1 \ , \\ \sigma_h(x_j) & , \quad \text{otherwise} \ , \end{cases} \tag{23}$$

$$\widetilde{u}_h|_{I_i} = \sum_{j=0}^{m+1} \bar{l}_j \widetilde{u}_j \ , \qquad \widetilde{u}_j = \begin{cases} \hat{u}_{i-\frac{1}{2}} & , \quad j = 0 \ , \\ \hat{u}_{i+\frac{1}{2}} & , \quad j = m+1 \ , \\ u_h(x_j) & , \quad \text{otherwise} \ , \end{cases} \tag{24}$$

It is convenient to define local vectors for the degrees of freedom in each cell:

$$u_i = (u_{i0}, \ldots, u_{i,m})^T \ , \tag{25}$$

$$\widetilde{u}_i = (\widetilde{u}_{i0}, \ldots, \widetilde{u}_{i,m+1})^T \ , \tag{26}$$

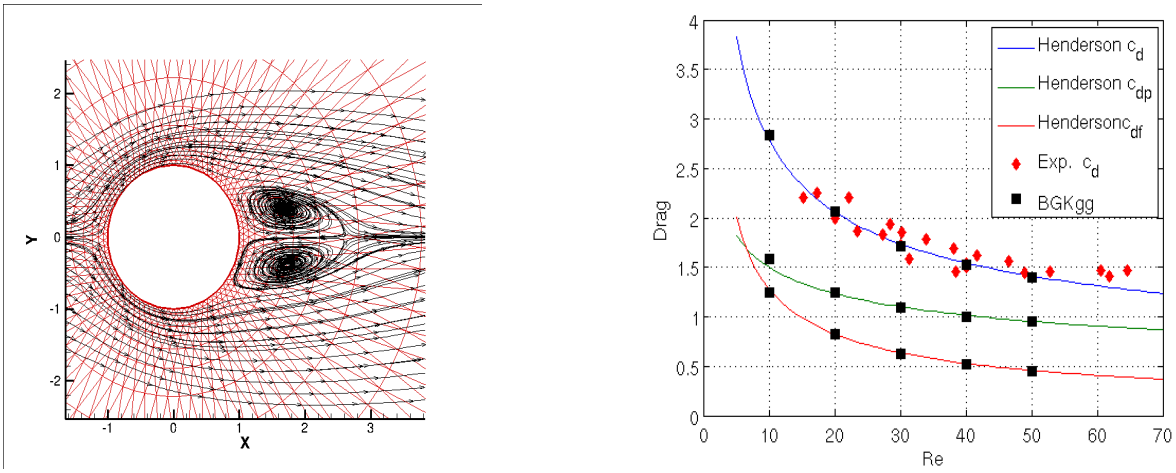
and similarly for the other local quantities $\widetilde{f}$, σ, and $\widetilde{\sigma}$. This leads to a scheme for each cell i:

$$\dot{u}_i + D(\widetilde{f}_i - \widetilde{\sigma}_i) = 0 \ , \tag{27}$$

$$\sigma_i - D^{\sigma} \widetilde{u}_i = 0 \ , \tag{28}$$

where $D \in \mathbb{R}^{m \times m+1}$ is a differentiation matrix with entries $d_{jk} = (2/\Delta x)\bar{l}'_k(\xi_j)$, and $D^{\sigma} \in \mathbb{R}^{m+1 \times m+1}$ is defined as $d^{\sigma}_{jk} = (2/\Delta x)\bar{l}'_k(\hat{\xi}_j)$. This formulation is quite generic in the sense that different numerical fluxes for the "elliptic" portion of the discretization may be employed, leading to nodal versions of such different schemes as the Local Discontinuous Galerkin method[14] or the simpler Bassi/Rebay discretization.[15] The extension to systems is quite simple. We refer to Cockburn and Shu[14] where such an extension is considered for the local DG scheme.

As an example for Navier-stokes equations, consider laminar steady flow around a cylinder. We have used the CUSP flux for the convective terms, and the simple Bassi/Rebay discretization[15] for the viscous discretization. Streamlines for this test case are shown in Fig. 1(a) for a Reynolds number Re=20. We have



(a) Streamlines for Re=20.

(b) Computational and experimental results from various sources

Figure 1. Laminar Flow around a Cylinder.

computed the flow for various Reynolds numbers, see Tab. 1, and recorded the drag coefficient. The flow was computed at comparatively high free-stream Mach number $M_\infty = 0.2$, due to the compressible formulation of the field equations. Nevertheless, the drag coefficient agrees well with other results. For comparison, see Fig. 1(b), where results using the kinetic BGKgg finite volume scheme,[6] numerical computations by Henderson[18] and experimental values[19,20] are plotted.

Re	c_d
20	2.07
30	1.77
40	1.58

Table 1. Drag Coefficient for laminar flow around a cylinder at $M_\infty = 0.2$ for various Reynolds numbers. Computations are performed with 4^{th} order Spectral Difference scheme.

III.B.1. Filtering

Aliasing is a well known issue with nodal high-order schemes.[21–23] The most popular countermeasure is filtering of the high frequency spectrum. The polynomial basis functions used with the Spectral difference scheme are not hierarchical, which necessitates an appropriate projection in order to access the high-frequency modes. A suitable change of basis is a simple matrix-vector operation. Consider the discrete solution

$$u_h = \sum_{j=1}^{N_m} u_j L_j = \sum_{j=1}^{N_m} \hat{u}_j \widetilde{\psi} \ , \tag{29}$$

where L_j are the Lagrange basis functions used in the Spectral Difference scheme, while the second sum is a representation in a suitably chosen modal basis $\widetilde{\psi}$. Evaluating eq. (29) at the solution interpolation nodes leads to

$$u = \mathcal{V}\hat{u} \ , \qquad \mathcal{V}_{ij} = \psi_j(\xi_i) \ , \tag{30}$$

where $\mathcal{V}$ is the generalized Vandermonde matrix corresponding to the basis ψ. Here we choose the Dubiner basis, a well-known modal orthogonal basis for the triangle.[24] Filtering may be applied by altering the coefficients of the modal basis as

$$\mathcal{F}(u_h) = \sum_{j=1}^{N_m} \sigma_j \hat{u}_j \widetilde{\psi} \ . \tag{31}$$

Here we apply the same filtering to coefficients of same total polynomial degree, of which there are $N_p - N_{p-1}$ for each $p \leq m$ (where we set $N_{-1} = 0$). Unfortunately this introduces some ad-hoc decisions into the methodology. The level of filtering one applies depends on the order of the scheme and the problem under consideration. Usually we apply filtering by attenuating the high frequency spectrum by a small amount (usually 5%) in the spirit of exponential filters. As an example consider Figure 2, where the Mach number contours for inviscid flow around the NACA0012 profile are shown. The figure shows a snapshot of the flow on the suction side of the airfoil. It can be seen that the filtering procedure removes artifacts that occasionally tend to manifest due to aliasing in underresolved regions.

IV. Relaxation Techniques

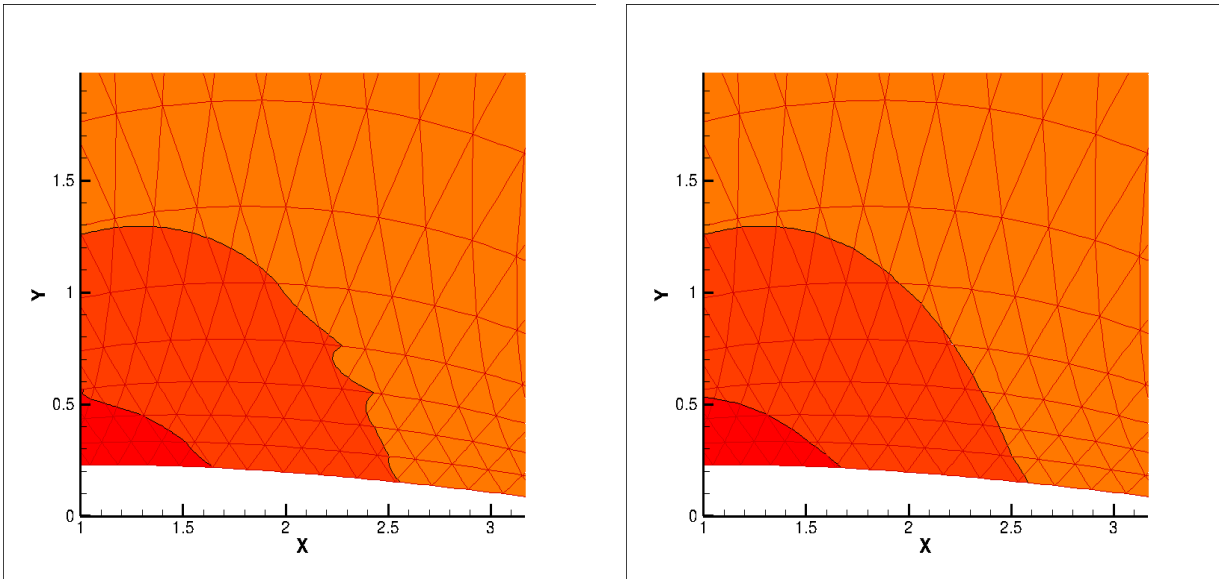
We focus here on the solution of the steady problem. By suitable spatial discretization the steady field equations are converted to a set of coupled algebraic equations

$$R(w_h) = 0 \ . \tag{32}$$

The core of the solution methodology pursued here is a relaxation in (pseudo-) time, marching the field equations to a steady state, considering the system of nonlinear ODE

$$\frac{dw}{dt^*} + R(w) = 0 \ . \tag{33}$$

The superscript $*$ is used as a reminder that no time accuracy is required to iterate toward the steady state, allowing such convergence acceleration techniques as local time stepping and multigrid methods.



(a) No Filtering

(b) Filtering of p=2 modes: $\sigma_j = 0.95\,,\ j = 4,5,6,\ \sigma_j = 1$ otherwise.

Figure 2. p_2 **Spectral Difference Scheme. Inviscid Flow around the NACA0012 profile at free-stream Mach number** $M_\infty = 0.3$ **and angle of attack** $\alpha = 0^o$**. Mach number contours on the suction side, approximately 25% to 75% chordlength.**

IV.A. Explicit Relaxation Methods with Multigrid

Explicit temporal discretization techniques of the Runge-Kutta type may be used for the ODE (33), which may be written

$$\begin{aligned} w^{(0)} &= w^n \\ w^{(k)} &= \sum_{l=0}^{k-1} \left\{ \alpha_{kl} w^{(l)} - \Delta t \beta_{kl} R^{(l)} \right\} \ , \qquad k = 1, \ldots, m \\ w^{n+1} &= w^{(m)} \ . \end{aligned} \tag{34}$$

Here we have used Shu's three-stage scheme,[25] which has been designed to preserve TVD properties of the spatial operator. Such TVD properties have been shown for the Spectral Difference Scheme with explicit time stepping in[6] using standard limiting methods. The three-stage scheme scheme has coefficients arranged in matrix form

$$\alpha = \begin{pmatrix} 1 & & \\ \frac{3}{4} & \frac{1}{4} & \\ \frac{1}{3} & 0 & \frac{2}{3} \end{pmatrix} \ , \qquad \beta = \begin{pmatrix} 1 & & \\ 0 & \frac{1}{4} & \\ 0 & 0 & \frac{2}{3} \end{pmatrix} \ . \tag{35}$$

Following Jameson,[26] geometric multigrid techniques may be combined with explicit Runge-Kutta methods, under the paradigm of the general FAS methodology.[27] Assume that the equations have been iterated n steps on a given mesh of characteristic length h, the "fine" mesh, and the residual $R_h(u_h^n)$ has been computed. After restricting solution as $u_{2h}^0 = I_h^{2h} u_h^n$, and likewise the residual, to a coarser mesh, advancing the solution on a coarse grid by one step can be accomplished by using the modified multistage scheme

$$u_{2h}^{(k)} = \sum_{l=0}^{k-1} \alpha_{kl} u_{2h}^{(l)} + \Delta t \beta_k \left(R_{2h}^{(k-1)} + S_{2h} \right) \ , \qquad k = 1, \ldots, m \ , \tag{36}$$

(37)

where

$$S_{2h} = I_h^{2h} R_h - R_{2h}^{(0)} \ . \tag{38}$$

This is formally identical to the original equation, except for an additional source term involving the residuals on the coarse and fine mesh. After relaxing on the coarse mesh for m iterations, the current solution $u^+_{2h} = u^m_{2h}$ is used to compute the corrected solution on the fine grid

$$u^+_h = u^n_h + I^{2h}_h (u^+_{2h} - u^0_{2h}) \ , \tag{39}$$

where I^{2h}_h is an interpolation operator. The extension to recursive application is obvious.

The extension of this methodology to discretizations of locally higher order, however, is not necessarily straight forward. For nonlinear hyperbolic equations, multigrid aids primarily through two mechanisms: Firstly the elimination of high-frequency error modes on successively coarser meshes, i.e. the same mechanism that has been demonstrated and analyzed extensively for elliptic equations. Secondly, the propagation of error modes, and expulsion from the computational domain is critically important. Here multigrid may be viewed simply as an increase of the effective wavespeed propagating these modes. This mechanism is absent in elliptic equations, but experience indicates that it is of critical importance for convection-dominated problems.

For higher order discretizations, on may let relaxation schemes of lower order approximation drive the solution on the highest level in the same way as described here for geometric multigrid. This is sometimes called $p-$multigrid. However, it is difficult to see, how the critical convective convergence mode is accelerated by this without still using coarser meshes as well. This leads us to believe that a successful method must necessarily include both components. Furthermore, the workload decreases only marginally for $p-$coarsening, which contributes to our view that $p-$multigrid is not necessarily very efficient. We instead opt for implicit methods for higher levels of polynomial approximation driven by geometric multigrid, as outlined in section IV.C.

IV.A.1. Stability and Robustness of Explicit Methods

While explicit relaxation methods are attractive due to ease of implementation and parallelization, stability restrictions are a concern. Linear and nonlinear stability properties of the Spectral Difference scheme with explicit methods have been analyzed in.[6,7] The spectrum of the discrete differentiation operator is proportional to m^2, where m is the polynomial order of approximation,[12] which suggests that stability for explicit methods necessitates $CFL \sim m^{-2}$. This has been confirmed for the one-dimensional linear advection equation, and permissible CFL numbers have been explicitly computed.[6] For convenience Tab. 2 reprints from[6] the unpublished permissible CFL numbers determined for the Spectral Difference Scheme. Here the Gauss-Legendre quadrature nodes corresponding to polynomials of degree m were used for flux interpolation, augmented with the two interval endpoints, yielding $m+2$ points. This allows flux interpolation of degree $m+1$, which leads to solution polynomial of degree m[a]. The Shu RK3 scheme, and Jameson's four-stage scheme[26] yield stable time integration. Results are compared with the standard DG method based on Legendre polynomials.

(a) SD / Jameson RK4

Order	cfl	cfl · DOF
1	0.696	1.392
2	0.363	1.089
3	0.226	0.904
4	0.156	0.780
5	0.115	0.690
6	0.089	0.623

(b) SD / Shu-RK3

Order	cfl	cfl · DOF
1	0.595	1.190
2	0.322	0.966
3	0.201	0.804
4	0.139	0.695
5	0.103	0.618
6	0.0799	0.559

(c) DG / Shu-RK3

Order	cfl	cfl · DOF
1	0.409	0.818
2	0.209	0.627
3	0.130	0.520
4	0.089	0.445
5	0.066	0.396
6	0.051	0.357

Table 2. Linear Stability Limits for Spectral Difference (SD) and Discontinuous Galerkin (DG) Schemes with Runge-Kutta Time Stepping. The right-most column displays the CFL number multiplied with the local degrees of freedom (DOF), which is appropriate when making comparison with standard finite difference schemes using the same number of total degrees of freedom.

It has to be stressed that very many combinations of explicit time integration methods with higher order schemes, such as the Spectral Difference scheme, or standard Discontinuous Galerkin schemes are not unconditionally stable at all. For example, for the 1D scheme, the popular Chebyshev-Lobatto nodes are not unconditionally stable[6,7] (and by extension tensor-products thereof). Linear instability has also been shown

[a]Note that in[6] these nodes had been erroneously labelled Legendre-Lobatto nodes

for the Spectral Difference scheme using different nodal sets for multivariate interpolation on triangular meshes.[7]

With these thoughts in mind, and anticipating very stiff problems for compressible viscous flow simulation at high Reynolds number, one ought to consider implicit relaxation methods as well.

IV.B. Implicit Relaxation

A backward Euler temporal discretization of (33) may be written

$$(I + \Delta t^* D_w R(w)|_{w^n})\, \Delta w^n = -\Delta t^* R(w^n) \;, \tag{40}$$

where $\Delta w^n = w^{n+1} - w^n$. For $\Delta t^* \to \infty$ one obtains a Newton iteration. Finite time steps may be interpreted as damped Newton iterations. We usually set $\Delta t \sim (\rho(D_w R))^{-1}$, where $\rho(A)$ is the spectral radius of the matrix A. The constant of proportionality is the implicit CFL number λ. We have not used very sophisticated time step control in the present work. A simple ramping method is introduced to avoid start-up problems:

$$\lambda = \begin{cases} z^2\,(3-2z)\,\lambda_{max} \;, & z < 1 \;, \\ \lambda_{max} \;, & \text{otherwise} \;, \end{cases} \tag{41}$$

with $z = \frac{N_{cyc}}{N_{start}}$, where N_{cyc} is the nonlinear iteration counter, and N_{start} is a fixed parameter, usually $N_{start} = 10$.

The key parameters are the implementation of the Jacobian matrix $A = D_w R$, and the solution methodology for the linear equations arising at each nonlinear iteration step n. (We defer the issue of pre-conditioning to section IV.B.4.) Here we have used the GMRES method.[28] The reason for this choice, besides favorable convergence properties, is the fact that one needs the Jacobian matrix only in action on the Krylov vectors, i.e. the matrix-vector products to generate the next Krylov vector $v^{(j+1)} = Av^{(j)}$. This facilitates the implementation of matrix-free methods considerably. Three different approaches are considered:

IV.B.1. Explicit assembly and storage of the exact Jacobian Matrix

The discrete field equations are a system of nonlinear algebraic equations. One may obtain the Jacobian by exact differentiation with respect to the solution vector $w = (w_1, \ldots, w_{N_{dof}})^T$, where N_{dof} is the total number of degrees of freedom, given by $N_{dof} = N_w N_m N_{elem}$. Here N_w is the number of equations ($N_w = 4$ in this case), N_m is the number of local degrees of freedom, see eq. (9), and N_{elem} is the number of mesh elements.

Computationally this is quite efficient. The overhead of assembling the matrix is mitigated by firstly the efficiency of the sparse matrix-vector product using the stored matrix elements, and secondly the possibility of freezing the Jacobian for a number of Newton iterations. We refer to this approach as "matrix-explicit".

There is a huge memory penalty on this approach. There are $(N_w N_m)^2 \cdot N_{elem}$ entries for the block-diagonal of the Jacobian alone. In two dimensions the storage requirement thus increases as m^4 (cf. eq. (9)), whereas in three dimensions we have $N_m \propto m^3$, and thus the required memory grows as m^6. For example, for $m = 6$ the size of the block diagonal is $(420 \times 420) \cdot N_{elem}$ for tetrahedral meshes (Note that $N_w = 5$ for three-dimensional flow) . If one supposes 8 bytes of storage for each entry this leads to 6.73 MB memory per mesh element, while for $m = 10$ one needs 78 MB for the diagonal block per element. When including off-diagonal elements and pre-conditioning the required memory increases by several factors. One can easily foresee situations where the memory requirement becomes prohibitive, or at least necessitates the use of massively parallel computations just to distribute memory, when the problem could be handled with more moderate computational resources in terms of floating point operations. It is thus advisable to consider matrix-free methods as well.

IV.B.2. Explicit assembly of the exact Krylov Vectors Av

The exact entries of the Jacobian matrices can be computed using efficient edge-based data structures. Instead of distributing the contributions of each degree of freedom to the proper memory location, one can instead sum the rows of the Jacobian with elements of the Krylov vector as a weight, i.e. $v_j^{(k+1)} = \sum a_{ij} v_j^{(k)}$, each time the matrix-vector product is needed. This is numerically equivalent to the matrix-explicit method of section IV.B.1 (not considering pre-conditioning for the moment), while necessitating only storage for

the Krylov vectors. However, the routines assembling the Jacobian need to be executed each time a new Krylov vector is generated during the GMRES iteration, which is computationally much more expensive than applying the sparse matrix vector product using the stored matrix elements. We have not considered this approach here.

IV.B.3. Numerical approximation of Krylov vectors

The matrix-vector product $R_w v$ is a projection of the total derivative of the residual onto the Krylov vector v. Hence one may generate a numerical approximation by

$$Av \approx \frac{R(w + \varepsilon v) - R(w)}{\varepsilon} , \tag{42}$$

where ε is a small parameter. Here the cost of applying the matrix-vector product is the same as one residual evaluation, which is also typically more expensive compared with the matrix-explicit approach in section IV.B.1, but much cheaper than the approach in section IV.B.2. However, there is additional uncertainty in the choice of ε. Several methods have been proposed in the literature to estimate the step size.[29,30] Here we adopt a rather simple method,[30] which may be written, supposing normalized Krylov vectors:

$$\varepsilon = \sqrt{1 + ||w||}\varepsilon_{rel} , \tag{43}$$

where the parameter $\varepsilon_{rel} = 10^{-6}$ was used in our implementation. We shall see below that this simple choice yields good results.

IV.B.4. Pre-conditioning

The primary obstruction against fully matrix-free implicit methods is the need to pre-condition the arising linear systems. Fully matrix-explicit methods using GMRES to solve the linear systems may be pre-conditioned using incomplete LU factorization,[31] denoted $ILU(n)$ where n stands for the level of additional fill allowed in the incomplete factorization. Generally low levels of fill are most efficient in terms of run time, compared to more complete factorizations that allow more effective pre-conditioning, but are significantly more expensive. We have used $ILU(n)$ pre-conditioning for all matrix-explicit methods.

For the matrix-free approach, the "flexible GMRES" formulation,[32] allows iterative solvers to be used as pre-conditioning methods. Recalling that the pre-conditioning matrix approximates A^{-1}, one generates a pre-conditioned Krylov vector by applying a few iteration steps with the same linear solver used in the Newton iteration. In particular, this may be done using the same matrix-free approximation. This method, which we denote "squared pre-conditioning", due to the recursive application of the linear solver, is used here for all matrix-free computations.

IV.C. Hybrid Multilevel Schemes

It is certainly possible to use multgrid with different relaxation schemes on different mesh levels and with different levels of approximation. In particular one may combine explicit and implicit techniques. Implicit and explicit schemes have very different constraints that may limit their usefulness on different mesh levels. Depending to the constraints deemed important one may find very different "optimal" combinations. For example, in[33] a scheme is proposed that uses a Discontinuous Galerkin method with an explicit Runge-Kutta discretization on the finest grid, with implicit schemes on the coarser level, with the primary concern being storage requirements on the finest grid. While storage requirements are certainly a very big concern, we are considering the exact opposite approach here. Explicit schemes are deemed ineffective for higher orders of accuracy, and $p-$multigrid not an optimal convergence acceleration, as discussed in section IV.A.

The main proposition of the present paper is a scheme that uses a damped Newton algorithm with GMRES linear solves on the highest level of approximation, while using explicit Runge Kutta smoothing with geometric h-Multigrid between Newton iterations using p_1 and lower levels of approximation. The rationale behind this is that experience indicates that error convection and expulsion is the primary mode of convergence, when considering integrated quantities, such as force coefficients, which are often essentially converged even at rather high levels of residuals, and when high-frequency errors still persist. For the acceleration of this convergence mode, no higher order solution representation is needed, and may indeed be

of hindrance. It seems attractive to use a fair number of explicit lower order multigrid cycles between each Newton step with p_m, $m > 0$ to accelerate the convection of large-scale error modes.

Intermediate p levels may be used through lower-order pre-conditioning instead of $p-$multigrid. In the present work, however, we used $ILU(n)$ pre-conditioning for matrix-explicit methods of all orders, while squared pre-conditioning is used for all matrix-free methods. The multilevel approach is illustrated in Fig. 3, where we have used the letter W to indicate a geometric multigrid $w-$cycle.

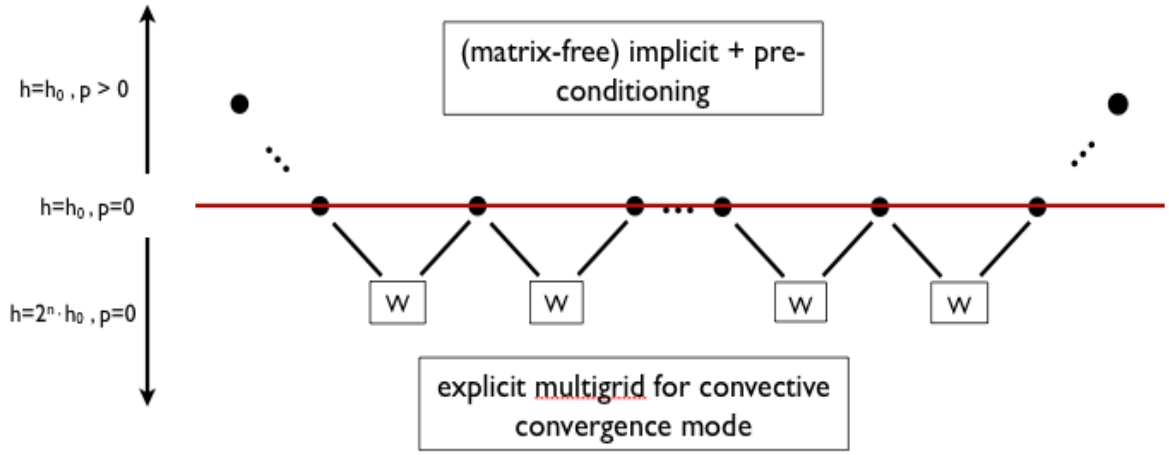


Figure 3. Illustration of hybrid multilevel relaxation strategy. The letter "w" indicates geometric multigrid with w-cycle explicit Runge-Kutta time stepping. Typically 10 or more explicit multigrid cycles are carried out between Newton iterations on the highest level of approximation.

V. Implementation

The implementation of the techniques described thus far can be tedious. Explicit multilevel methods have been implemented previously.[6] Thus the main burden is on implementing the implicit relaxation techniques, such as derivation of the exact Jacobian, and implementation of different solution methods, including pre-conditioning techniques. We used the PETSc library,[34] which is particularly useful for matrix-explicit methods, removing the implementation of pre-conditioning. For the matrix-free methods, user-defined routines implementing the matrix-vector product Av have to be provided. In this case, the usefulness of the PETSc library is limited to driving the GMRES algorithm, calling in turn the necessary user-defined routines.

V.A. Using Automatic Differentiation

In the one-dimensional case the block-diagonal of the system Jacobian consists of $m+1 \times m+1$ blocks of 3×3 matrices, which can be written for the j^{th} local row and the n^{th} local column in the block corresponding to a particular cell

$$\frac{\partial R_j}{\partial w_n} = \sum_k d_{jk} \widetilde{A}_k \, l_{kn} \ , \qquad j,n = 0,\ldots,m \ , \tag{44}$$

where d_{jk} is the local differentiation matrix $d_{jk} = J^{-1}\hat{l}'_k(\xi_j)$, and the reconstruction coefficients are given by $l_{kn} = l_n(\hat{\xi}_k)$. The matrix $\widetilde{A}$ is given by the flux Jacobian evaluated at the flux collocation points for interior nodes, and the differentiation of the numerical flux function h for nodes on element boundaries. It can be seen that assembly of the Jacobian is relatively straight-forward, since the reconstruction and differentiation matrices are available for the residual computation.

In higher dimensions the structure of the Jacobian remains the same, with the only difference that d sums are needed, and the matrices $\widetilde{A}$ become appropriately transformed matrices, due to the mapping Φ. The only difficulty lies in the differentiation of the numerical flux h. This is the only part of the system Jacobian that is not generic in the sense that a change of the numerical flux function requires a non-trivial re-derivation of the Jacobian matrix.

In order to easily generate these derivatives, we adopted TAPENADE[b], an automatic differentiation (AD) engine developed at the INRIA Sophia-Antipolis research centre. TAPENADE can be utilized as a server, but can also be installed locally and be run by a simple command line, which can be included into a Makefile. We actually adopted this second solution, so that no additional manual operations are required by the user in case of implicit time integration. There is no need for writing a new routine which computes the derivatives of the numerical flux: exchanging the numerical flux routine, recompiling and executing the program will still be the only required operations.

After installing TAPENADE locally we had to transform the subroutine which computes the numerical flux into a function, having among its arguments the dependent variables (i.e., the fluxes), which we want to differentiate, and the independent variables (i.e., the conservative variables), w.r.t. which we want to differentiate. The `tapenade` command line looks as follows:

```
tapenade -tangent -head numerical_flux -outvars "left_flux right_flux"
   -vars "left_density left_x_momentum left_y_momentum left_energy
   right_density right_x_momentum right_y_momentum right_energy"
   -outputfiletype fortranplushtml numerical_flux.f90
```

This command can be introduced into the Makefile as a target rule, where the source code to be differentiated is given at the end of the `tapenade` command (i.e., *numerical_flux.f90*).

By inserting this instruction in the solver's Makefile, we achieve our aim: every time that the routine computing the numerical flux (*numerical_flux.f90* in this example) is modified, TAPENADE is automatically called and the differentiated routine (*numerical_flux_d.f90*) for the Jacobian computation is newly generated. This makes our solver extendable to any kind of numerical flux and the update requires no further job from the user's point of view and little extra work from the computational point of view.

This procedure shows a few disadvantages. Firstly, the differentiated routine may not be the optimal one. If for instance one of the dependent variables does not depend on one of the independent variables, zeros will be computed and carried on along the computation, with consequent waste of computational time. However, if interested in optimizing the solving procedure, one could use AD to create the differentiated routine and then manually optimize that routine later. Secondly, having TAPENADE installed locally means that from time to time one has to re-install to get updates of the tool. In our opinion however the enhancement provided to our solver by the adoption of AD is worth accepting these small disadvantages.

VI. Numerical Experiments

We first carry out a study with different orders of approximation using matrix-explicit and matrix-free methods.

Consider inviscid flow over a smooth bump at free-stream Mach number $M_\infty = 0.3$ (see Fig. 4)). Computations were performed using the Spectral Difference scheme on a regular triangular mesh with 342

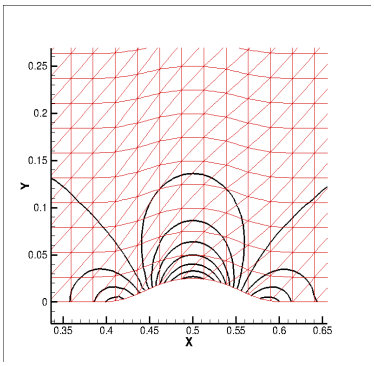
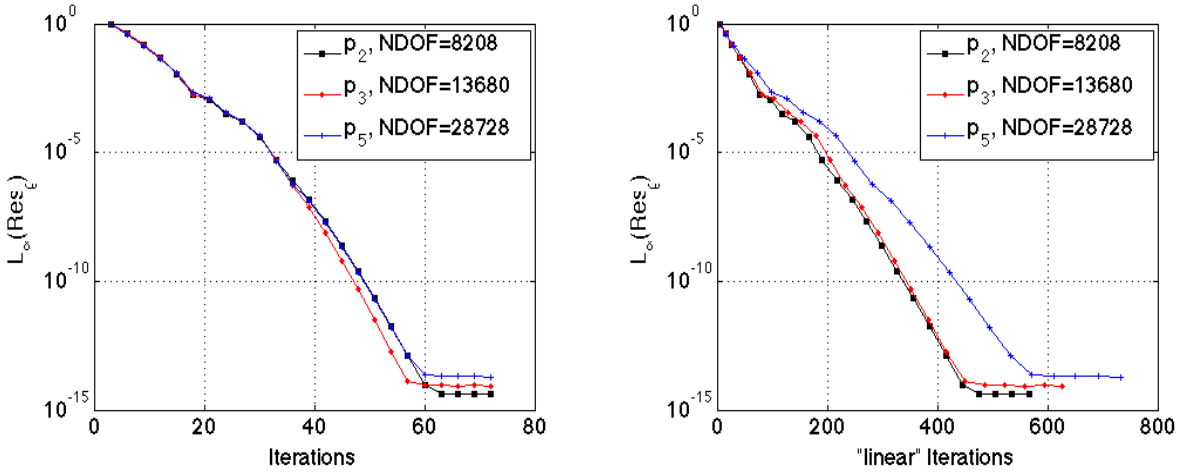


Figure 4. Contours of constant Mach number for inviscid flow over a smooth bump at free-stream Mach number $M_\infty = 0.3$. Computation performed with the Spectral Difference scheme using cubic polynomials ($m = 3$).

cells. Polynomial orders of $m = 2$, $m = 3$, and $m = 5$ have been used, leading to a maximum number of degrees of freedom of $N_{dof} = N_w N_m N_{elem} = 28,728$ for the case $m = 5$. A comparison of the convergence for various orders of approximation is shown in Fig. 6 for the matrix-explicit method. The Jacobian matrix

[b]TAPENADE can be downloaded at http://www-sop.inria.fr/tropics/tapenade.html



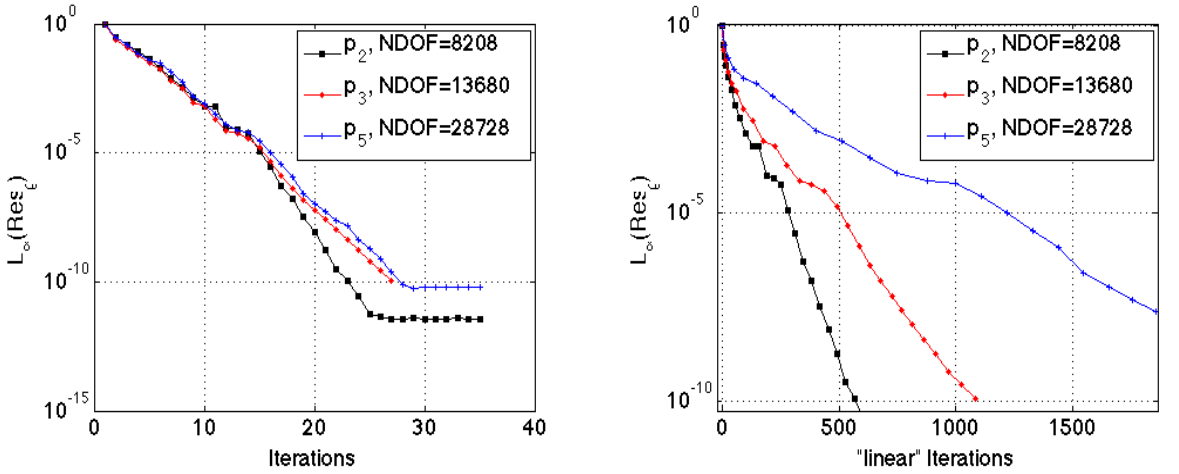
(a) Density Residual vs. Newton Iterations

(b) Density Residual vs. cumulative GMRES Iterations

Figure 5. Convergence of the matrix-explicit method for smooth inviscid flow around a bump at free-stream Mach number $M_\infty = 0.3$.

of the system is based on an exact differentiation of the residual, including the convective flux function, in this case Jameson's CUSP flux,[13] and is re-computed every three Newton iterations. For the case $m = 2$ we use $ILU(3)$ pre-conditioning, while $ILU(4)$ is used for the other cases. by default GMRES is restarted after 30 iterations. Besides the convergence in terms of Newton iterations (Fig. 5(a)) we also plot the convergence against cumulative GMRES iterations, i.e. number of Krylov vectors generated. Note that the number of linear iterations shown in Fig. 5(b) is by itself not very meaningful, as it can be adjusted very easily through pre-conditioning. In particular, if an exact LU decomposition is used this number becomes equal to the number of nonlinear Newton iterations, at the expense of much increased computational cost. The GMRES iterations are shown here merely to give an idea of representative "best practice" results, and facilitate comparison between different approaches.

Consider the same test case with the matrix-free implementation outlined in section IV.B.3 using squared pre-conditioning as described in section IV.B.4. The convergence is shown in Fig. 6. The number of Newton iterations needed to drive the solution to a steady state is not significantly affected by the matrix-free



(a) Density Residual vs. Newton Iterations

(b) Density Residual vs. cumulative GMRES Iterations

Figure 6. Convergence of the matrix-free method for smooth inviscid flow around a bump at free-stream Mach number $M_\infty = 0.3$.

approach. This indicates that numerical approximation of the Krylov vector (cf. eq. (42)) yields consistently accurate results, despite the relatively simple step size estimation (43) in the difference approximation. In fact, compared with the best practice matrix-explicit approach, which is usually more efficient when saving the Jacobian for a number of iterations, the Newton iteration count is lower. Here the matrix elements

cannot be saved, and instead the numerical approximation (42) is carried out each time a new Krylov vector is needed, which decreases the number of nonlinear iterations, because a more up-to-date approximation of the Jacobian is used, but also leads to more residual evaluations. The convergence in terms of linear iterations is typically slower. This stems from the fact that it is usually advisable to keep the number of linear iterations carried out for pre-conditioning purposes relatively low in order to avoid the associated computational overhead. This means that one has to tolerate a higher number of GMRES iterations in each Newton cycle.

We now turn to the hybrid multilevel method outlined in section IV.C. As an example consider smooth flow around the NACA0012 profile, see Fig. 7, at free-stream Mach number $M_\infty = 0.3$. We use a third

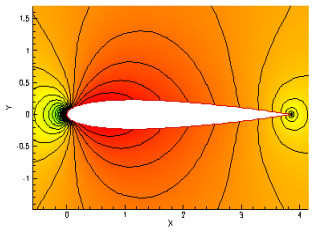


Figure 7. Contours of constant Mach number for inviscid flow around the NACA0012 profile at free-stream Mach number $M_\infty = 0.3$ and angle-of-attack $\alpha = 0^o$.

order accurate method, i.e. quadratic polynomials, on the highest level of approximation, along with an implicit damped Newton/GMRES method. For now we consider the matrix-explicit approach with $ILU(2)$ pre-conditioning. Between each Newton step 20 multigrid cycles with explicit RK smoothing are performed (cf. Fig. 3), starting from a piecewise linear finite-volume method on the same mesh, followed by a first order finite-volume method on two coarser meshes. The meshes used for this test case are summarized in Tab. 3. Besides the number of elements and the total number of degrees of freedom we show the level of approximation and the cfl number used on each level. The explicit multigrid procedure carried out between levels 3-1 uses the same relaxation method on all levels, given by the three-stage Runge-Kutta method due

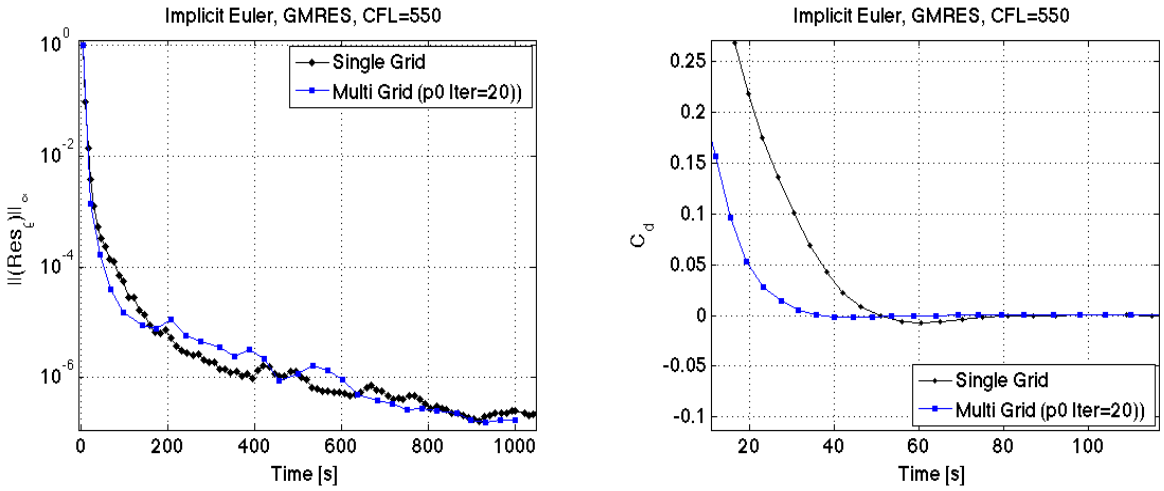
Hybrid Multilevel					
Level	DOF	p	Cells	cfl	Type
4	15360	2	2560	550	Implicit
3	2560	1	2560	6	Explicit
2	640	0	640	6	Explicit
1	160	0	160	6	Explicit

Table 3. Meshes and Degrees of freedom (DOF) used with the hybrid multilevel method.

to Shu[35] presented in section IV.A.

The benefit of using the explicit relaxation steps between each Newton iteration comes from the fact that these come at virtually no cost, compared to the implicit cycles. The speedup afforded by this approach is demonstrated in Fig. 8. Here the multilevel method is compared to an implicit Newton method identical to the approximation level 4, but with no additional explicit relaxation steps. For this coarse mesh the residual reduction rate is almost unchanged, when plotted against execution time. Here the speed-up is roughly offset by the additional cost. However, the convergence of the drag is much improved. Again it is clearly visible that multigrid has a more dramatic effect on convergence of integrated quantities, such as force coefficients, than residual reduction rate.

Consider now the comparison of the hybrid multilevel method with a straight explicit h/p multigrid method for the same test case, where we use the RK3 scheme on the highest level of approximation. The h/p-multigrid strategy is summarized in Tab. 4. In a sense we simply trade off the severe stability restriction for the explicit scheme with the much reduced cost per iteration. Note that for the explicit h/p multigrid method we use only one w-cycle per iteration on the finest mesh. Naturally there is a lower iteration count for the hybrid solution method compared to the explicit method. However, this does not necessarily translate into a faster turnaround. Figure 9 shows a plot of the density residuals, plotted against both iteration number and execution time. It can be seen that while the reduction rate per cycle is clearly higher for the



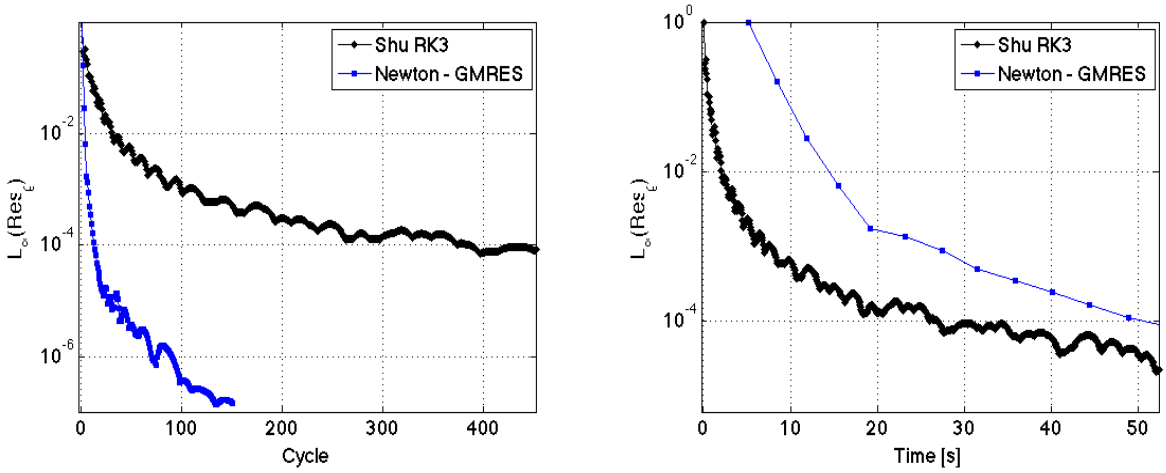
(a) Density Residual vs. Time

(b) Drag Coefficient vs. Time

Figure 8. Inviscid Flow around the NACA0012 profile: Comparison of implicit relaxation with and without additional geometric multigrid smoothing.

Explicit h/p Multigrid				
Level	DOF	p	Cells	cfl
4	15360	2	2560	1
3	2560	1	2560	6
2	640	0	640	6
1	160	0	160	6

Table 4. Meshes and Degrees of freedom (DOF) used with h/p-multigrid.

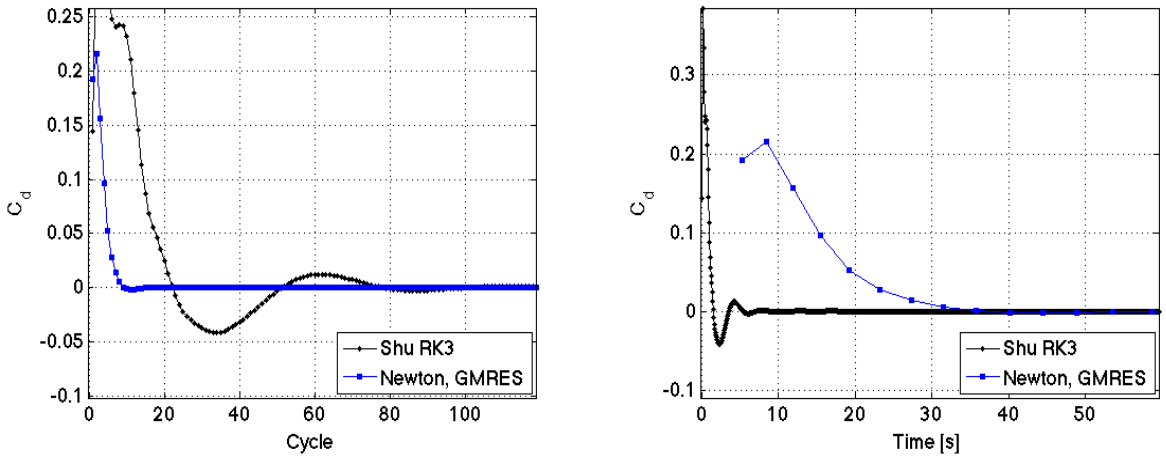


(a) Density Residual vs. Cycles

(b) Density Residual vs. Time

Figure 9. Inviscid Flow around the NACA0012 profile: Comparison of convergence for explicit h/p-multigrid and hybrid multilevel scheme

best-practice implicit method, the explicit multigrid method is faster measured in CPU time. This becomes even more visible when comparing the convergence for the drag coefficient, which is depicted in Fig. 10 for both measures, cycle count and time. Thus it seems that here the method of choice is explicit time-stepping



(a) Coefficient of Drag vs. Cycles

(b) Coefficient of Drag vs. Time

Figure 10. Inviscid Flow around the NACA0012 profile: Comparison of convergence for explicit h/p-multigrid and implicit schemes

with h/p-Multigrid. Even though the implicit solution methodology greatly reduces the number of cycles needed to converge to a steady state, the efficiency per cycle of the explicit method wins out. However, this is not likely to extend to higher order approximations, or high-Reynolds number viscous flow, as the stability restrictions become very severe for the explicit method. Furthermore, the reduction in computational cost compared to the highest level of polynomial approximation is by far less dramatic for $p-$multigrid, compared with traditional $h-$multigrid, based on mesh coarsening. At present transfer operators for higher levels of $p-$multigrid are not implemented. We defer investigation of explicit h/p-multigrid for higher orders to a future publication.

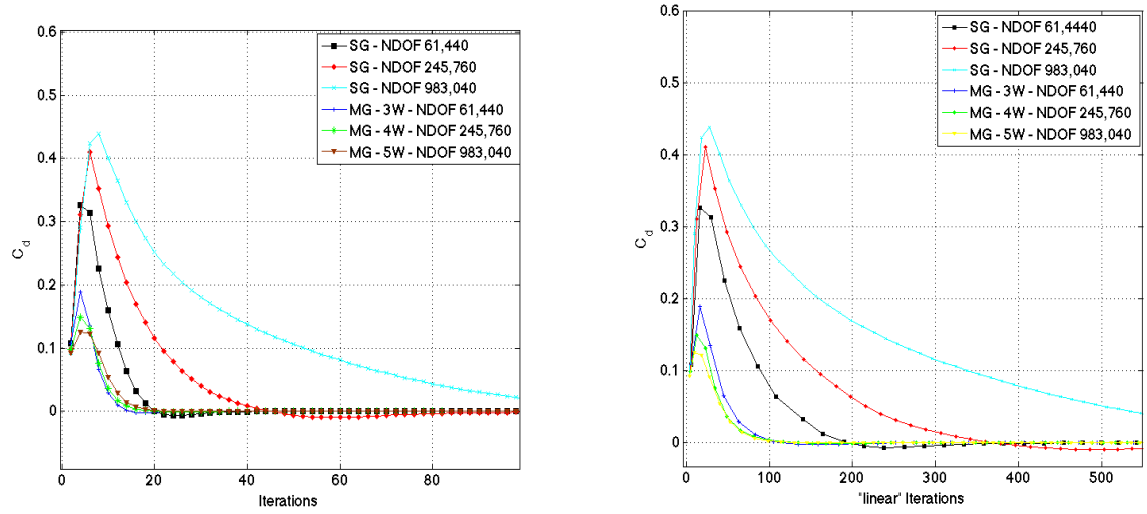
We now turn to examining the convergence properties in mesh refinement. The aim of geometric multigrid methods has traditionally been to achieve mesh-independent convergence rates. For the NACA0012 test case we consider three different meshes, which are summarized in Tab. 5. The coarsest of these is identical to the finest mesh in Tab. 3, and uses the same multigrid strategy. The multigrid strategy for a finer mesh includes

Level	N_{dof}	Elements
fine	983,040	40960
medium	245,760	10240
coarse	61,440	2560

Table 5. Meshes used for h-refinement study for hybrid Multilevel method.

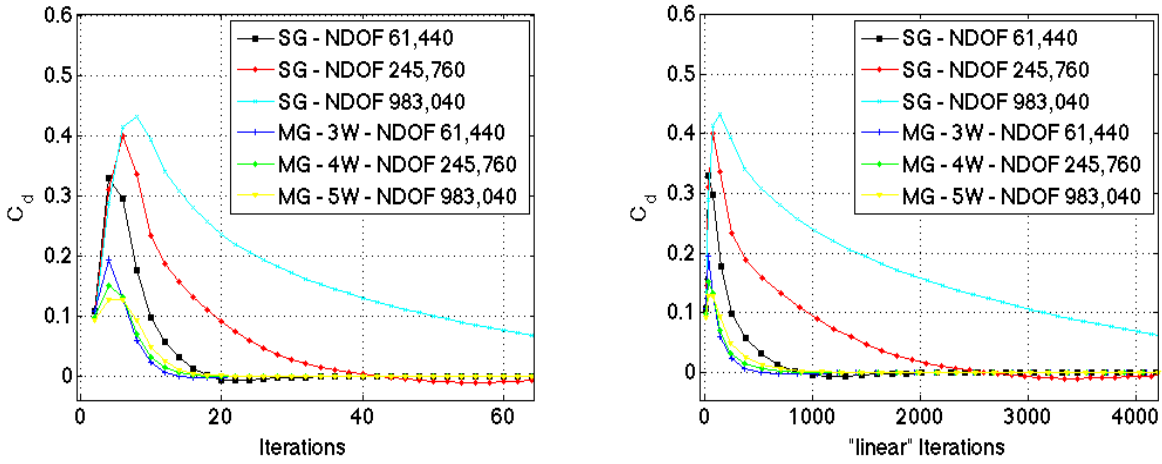
the next coarser one recursively, so that the coarsest mesh is always the same (cf. Tab. 3), and the total number of meshes used increases by one for every refinement. Consider first the matrix-explicit approach. We use $ILU(3)$ pre-conditioning for the implicit method on the highest level of approximation. Figure 11 shows the convergence of the drag coefficient vs. Newton iterations and cumulative GMRES iterations. It can be seen that in both measures the convergence of the multigrid method is very nearly mesh independent. In comparison the convergence of the single-grid implicit iterations that use the same methodology as the multilevel method on the finest mesh degrades as the mesh is refined. It is remarkable that the benefit of the additional multigrid relaxations come at trivial computational expense, but aid tremendously in the convergence process.

Consider in comparison the matrix-free version applied to the same mesh refinement study. As shown in Fig. 12 the convergence is still nearly mesh-independent for the multilevel method with the matrix-free implementation. Again the number of Newton iterations is virtually unchanged, indicating good quality of numerical approximation of the Jacobian. It should be stressed that the memory savings become extreme for the finer meshes considered here. The finest mesh uses several GB worth of memory with the matrix-



(a) (b)

Figure 11. Inviscid Flow around the NACA0012 profile at free-stream Mach number $M_\infty = 0.3$, angle-of-attack $\alpha = 0^0$



(a) Drag coefficient vs. Newton iterations (b) Drag coefficient vs. accumulated GMRES iterations

Figure 12. Inviscid Flow around the NACA0012 profile at free-stream Mach number $M_\infty = 0.3$, angle-of-attack $\alpha = 0^0$.

explicit approach (for a 2D computation!). In fact, a further doubling of the mesh size was not possible using the matrix explicit approach due to memory limitations on a machine with 8GB total memory. This should motivate pursuing matrix-free methods, certainly for three-dimensional computations, where storage requirements increase as m^6 instead of m^4.

VII. Future Work

It can be expected that for higher orders of approximation the implicit method should outperform the explicit h/p-multigrid method even for inviscid flow, considering that permissible CFL numbers decrease with m^{-2}. However, this needs to be confirmed. For this purpose the explicit method needs to be extended to higher order of accuracy using appropriate transfer coefficients for the $p-$multigrid method.

We have mentioned that the primary obstacle one faces when implementing matrix-free methods is pre-conditioning. So far squared pre-conditioning has been used. Alternatively lower order pre-conditioning may be used, which perhaps yields comparable results to squared pre-conditioning at reduced computational expense.

The comparison between different relaxation methods has focused on inviscid flow in this paper. We are currently in the process of extending the automatic generation of Jacobian matrices to the viscous case. Once this is complete a more thorough investigation into viscous flow should be carried out. The implicit method can be expected to become more efficient compared with the explicit approach as the Reynolds-number increases, due to mesh-induced stiffness when resolving boundary layers with anisotropic meshes.

VIII. Acknowledgements

Financial support from the Deutsche Forschungsgemeinschaft (German Research Association) through grant GSC 111, and from European Office of Aerospace Research and Development through grant FA8655-08-1-3060 is gratefully acknowledged.

References

[1]Liu, Y., Vinokur, M., and Wang, Z. J., "Discontinuous spectral difference method for conservation laws on unstructured grids," *Proceedings of the 3rd International Conference on Computational Fluid Dynamics, July 12-16, 2004, Toronto, Canada*, Springer, 2004.

[2]Liu, Y., Vinokur, M., and Wang, Z. J., "Spectral Difference method for unstructured grids I: Basic formulation," *J. Comp. Phys.*, Vol. 216, No. 2, 2006, pp. 780–801.

[3]Wang, Z. J., Liu, Y., May, G., and Jameson, A., "Spectral Difference Method for Unstructured Grids II: Extension to the Euler Equations," *J. Sci. Comput.*, Vol. 32, No. 1, 2007, pp. 54–71.

[4]Kopriva, D. A. and Kolias, J. H., "A Conservative Staggered-Grid Chebyshev Multidomain Method for Compressible Flows," *J. Comp. Phys.*, Vol. 125, 1996, pp. 244–261.

[5]May, G., "The Spectral Difference Scheme as a Quadrature-Free Discontinuous Galerkin Method," submitted to Journal of Computational Physics, AICES Technical Report 2008-11, Aachen Institute for Advanced Study in Computational Engineering Science, http://www.aices.rwth-aachen.de/preprints, 2008.

[6]May, G., *A Kinetic Scheme for the Navier-Stokes Equations and High-Order Methods for Hyperbolic Conservation Laws*, Ph.D. thesis, Stanford University, Stanford, CA 94305, 2006.

[7]van den Abeele, K., Lacor, C., and Wang, Z. J., "On the Stability and Accuracy of the Spectral Difference Method," *J. Sci. Comput.*, Vol. 37, No. 2, 2008, pp. 162–188.

[8]Blyth, M. G. and Pozrikidis, C., "A Lobatto Interpolation Grid over the Triangle," *IMA J. Appl. Math.*, Vol. 71, 2006, pp. 153–169.

[9]Chen, Q. and Babuška, I., "Approximate Optimal Points for Polynomial Interpolation of Real Functions in an Interval and in a Triangle," *Comput. Meth. Appl. Mech. Eng.*, Vol. 128, No. 3-4, 1995, pp. 405–417.

[10]Hesthaven, J. S., "From Electrostatics to Almost Optimal Nodal Sets for Polynomial Interpolation in a Simplex," *SIAM J. Num. Anal.*, Vol. 35, No. 2, 1998, pp. 655–676.

[11]Taylor, M. A., Wingate, B. A., and Vincent, R. E., "An algorithm for computing Fekete points in the triangle," *SIAM J. Numer. Anal.*, Vol. 38, No. 5, 2000, pp. 1707–1720.

[12]Hesthaven, J. S. and Gottlieb, D., "Stable Spectral Methods for Conservation Laws on Triangles with Unstructured Grids," *Comput. Meth. Appl. Mech. Engrg.*, Vol. 175, 1999, pp. 361–381.

[13]Jameson, A., "Analysis and design of numerical schemes for gas dynamcics 2: Artificial diffusion and discrete shock structure," *Int. J. Comp. Fluid. Dyn.*, Vol. 5, 1995, pp. 1–38.

[14]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.

[15]Bassi, F. and Rebay, S., "A High-Order Accurate Discontinuous Finite-Element Method for the Numerical Solution of the Compressible Navier-Stokes Equations," *J. Comp. Phys.*, Vol. 131, 1997, pp. 267–279.

[16]Lomtev, I. and Karniadakis, G. E., "A Discontinuous Galerkin Method for the Navier-Stokes Equations," *Int. J. Numer. Meth. Fluids*, Vol. 29, 1999, pp. 587–603.

[17]Arnold, D. N., Brezzi, F., Cockburn, B., and Marini, L. D., "Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems," *SIAM J. Num. Anal.*, Vol. 39, No. 5, 2002, pp. 1749–1779.

[18]Henderson, R. D., "Details of the Drag Curve Near the Onset of Vortex Shedding," *Phys. Fluids*, Vol. 7, No. 9, 1995, pp. 2102–2104.

[19]Takami, H. and Keller, H. B., "Steady Two-Dimensional Viscous Flow of an Incompressible Fluid Past a Circular Cylinder." *Phys. Fluids*, Vol. 12, No. II, 1969, pp. 51–56.

[20]Tritton, D. J., "Experiments on the Flow Past a Circular Cylinder at Low Reynolds Numbers," *J. Fluid Mech.*, Vol. 6, 1959, pp. 547–567.

[21]Boyd, J. P., *Chebyshev and Fourier Spectral Methods*, Dover, 2nd ed., 2000.

[22]Gottlieb, D. and Hesthaven, J. S., "Spectral Methods for Hyperbolic Problems," *J. Comp. Appl. Math*, Vol. 128, No. 1-2, 2001, pp. 83–131.

[23]Hesthaven, J. S. and Warburton, T., *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications*, No. 54 in Texts in Applied Mathematics, Springer Verlag, 2007.

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

[25]Shu, C.-W. and Osher, S., "Efficient Implementation of Essentially Non-Oscillatory Shock Capturing Schemes," *J. Comp. Phys.*, Vol. 77, 1988, pp. 439–471.

[26]Jameson, A., "Solution of the Euler Equations For Two Dimensional Transonic Flow by a Multigrid Method," *Appl. Math. Comp.*, Vol. 13, 1983, pp. 327–356.

[27]Brandt, A., "Multi-Level Adaptive Solutions to Boundary-Value Problems," *Math. Comp.*, Vol. 31, No. 138, 1977, pp. 333–390.

[28]Saad, Y. and Schultz, M. H., "GMRES: A Generalized Minimal Residual Algorithm for Solving Non-Symmetric Linear Systems," *SIAM J. Sci. Stat. Comp..*, Vol. 7, 1986, pp. 856–869.

[29]Brown, P. N. and Saad, Y., "Hybrid Krylov methods for nonlinear systems of equations," *SIAM J. Sci. Stat. Comput.*, Vol. 11, 1990, pp. 450–481.

[30]Pernice, M. and Walker, H. F., "NITSOL: A Newton iterative solver for nonlinear systems," *SIAM J. Sci. Stat. Comput.*, Vol. 19, 1998, pp. 302–318.

[31]Saad, Y., *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, 2nd ed., 2003.

[32]Soulaimani, A., Salah, N. B., and Saad, Y., "Enhanced GMRES Acceleration Techniques for some CFD Problems," *Int. J. Comp. Fluid. Dyn.*, Vol. 16, No. 1, 2002, pp. 1–20.

[33]Luo, H., Baum, J. D., and Löhner, R., "A p-Multigrid Discontinuous Galerkin Method for the Euler equations on unstructured grids," *J. Comp. Phys.*, Vol. 211, 2006, pp. 767–783.

[34]Balay, S., Buschelman, K., Eijkhout, V., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Smith, B. F., and Zhang, H., "PETSc Users Manual," Tech. Rep. ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, 2004.

[35]Shu, C. W., "Total-Variation-Diminishing Time Discretizations," *SIAM J. Sci. Stat. Comput.*, Vol. 9, No. 6, 1988, pp. 1073–1084.