Continuous and Discontinuous Galerkin Methods for Atmospheric Modeling

Francix X. Giraldo

Department of Applied Mathematics
Naval Postgraduate School
Monterey, CA 93943 USA

ABSTRACT

In this paper, we describe continuous (CG) and discontinuous Galerkin (DG) methods for atmospheric modeling. We present a unified CGDG formulation for general nonlinear systems of hyperbolic equations. We then describe the possible mechanisms available to both CG and DG for maintaining positivity of, e.g., tracers such as those representing moist variables. In addition, we include a discussion on some of the strengths of CGDG methods such as (non-conforming) adaptive mesh refinement and the exceptional scalability of CGDG methods on massively parallel computers.

1 Introduction

The goal of the work being conducted at the Department of Applied Mathematics of the Naval Postgraduate School is to construct a nonhydrostatic atmospheric model that has the following characteristics: highly scalable on current (CPU-based) and future (hybrid CPU- and GPU-based) computer architectures; permitting general grids such as statically and dynamically adaptive; highly efficient; high-order accuracy in all the numerics (both space and time); unified regional and global numerical weather prediction; and conservative, minimally dissipative with excellent dispersion properties. However, it should be emphasized that the focus of this paper will remain mostly within the area of spatial discretization - that is, how to approximate the spatial derivatives. There are many possibilities to choose from to approximate the spatial derivatives but whichever method is chosen should have the following properties: stability and robustness, accuracy, efficiency, geometric flexibility, and long shelf-life. It is clear that stability is paramount but robustness is a bit more difficult to understand. Here, we write it to mean that the method remains stable across a large range of applications and additions to either physical or numerical processes. One example regarding numerical processes could mean how well does a method handle variable resolution or non-conforming grids. We generally do not know what is the *sweet spot* for the order of accuracy in a method but it is easy to argue that high order is preferable to low order although the efficiency of the method has to be taken into account. Geometric flexibility means that the method is capable of handling all levels of refinement. For example h-refinement means that we automatically decrease the mesh spacing size (h), while r-refinement means that we move the grid points around, and p-refinement means that we change the order of the approximation within a grid box whenever a certain threshold is reached. The reason why geometric flexibility is important has to do with increasing the shelf-life of a model. For example, once nonhydrostatic models reach fine scale resolutions it may be possible to perform simulations previously not realizable. Even at the exascale range of computing it may be necessary to use mesh refinement to run, e.g., high-resolution hurricane/storm-surge simulations. If a numerical method is not equipped to handle mesh refinement then models based on these numerical methods may have to be rewritten in a few years after operations. Additionally, if methods are capable of being used on one set of computers but not others then a change in the computer industry will require rewriting models. We have seen examples of this with global methods that were ready-made

for vector computers but more difficult (although not impossible) to adapt to distributed computers and now Graphics Processing Units (GPUs). Since it is expected that data transfer will remain the bottleneck within the foreseeable future, methods based on local stencils will reign supreme.

The remainder of the paper is organized as follows. In Sec. 2 the unified presentation of the continuous Galerkin (CG) and discontinuous Galerkin (DG) methods is described for the general class of systems of nonlinear hyperbolic equations. In Sec. 3 we discuss some possibilities for preserving positivity in both CG and DG methods. The Nonhydrostatic Unified Model of the Amtosphere (NUMA) is briefly described in Sec. 4 because it is the model used for developing and studying CGDG methods. Section 5 describes the non-conforming adaptive mesh refinement approach used with both CG and DG methods. Then in Sec. 6 we describe the parallelization strategy for both CG and DG and compare their results on tens of thousands of cores. This section also describes the on-going effort to port CGDG methods (within the NUMA model) to GPUs. In Sec. 7 we present a summary of our findings and discuss directions for future work.

2 Unified Continuous and Discontinuous Galerkin Methods

High-order continuous Galerkin (CG) methods were first proposed for the atmosphere by Taylor et al. [15] in 1997 for the shallow water equations on the sphere. Less than a decade later, the first high-order CG hydrostatic models began appearing [7, 2]. Discontinuous Galerkin (DG) methods came later. DG methods were first proposed for the atmosphere by Giraldo et al. [5] in 2002 for the shallow water equations on the sphere. One of the main contributions of that work is that the authors essentially proposed a discontinuous version of the spectral element method so that the DG method was now able to compete, in terms of efficiency, with spectral element methods. Since that paper, we have seen both hydrostatic [14] and nonhydrostatic [6, 9] models come on line.

To allow the description of the CG and DG methods to remain as general as possible, let us derive the unified CGDG method for the system of nonlinear (non-homogeneous) conservation laws

$$\frac{\partial q}{\partial t} + \nabla \cdot F = S(q) \tag{1}$$

where, for example, q is the state vector which could represent $(\rho, u^{\mathscr{T}}, \theta)^{\mathscr{T}}$ with ρ being the density, $u = (u,v,w)^{\mathscr{T}}$ the Cartesian velocity field, and θ the potential temperature. Other items requiring definition include the ∇ which is the operator $\left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right)^{\mathscr{T}}$, F is the flux tensor, and $S(q)$ contains the source terms which make Eq. (1) non-homogeneous.

2.1 Element-based Galerkin Methods

To discretize Eq. (1) in space via CGDG methods we first represent the global domain as a collection of non-overlapping control volumes. For those familiar with either finite element or finite volume methods, these *building blocks* of the discretization are known as either elements, volumes, or cells. An example of this strategy is depicted in Fig. 1 where we show that inside each control volume, many more degrees of freedom could be defined in order to increase the order of approximation.

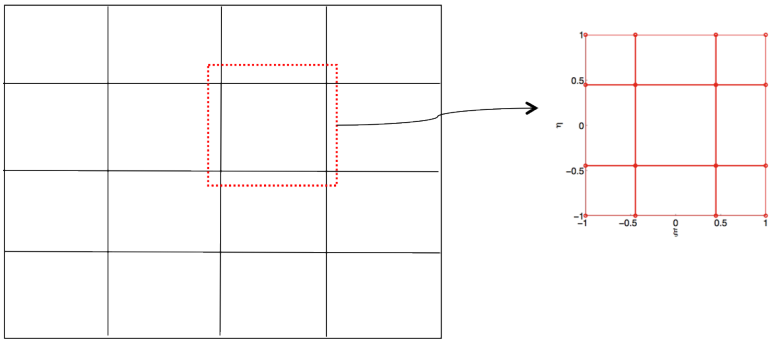
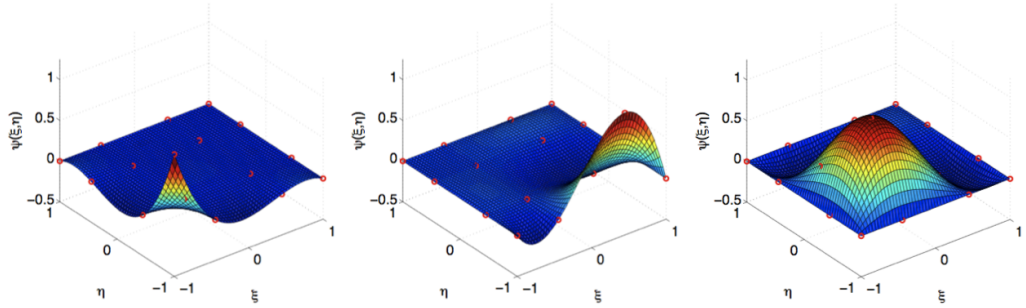


Figure 1: Element-based Galerkin Methods: the global domain is subdivided into a collection of non-overlapping cells (left). Inside each cell one can define multiple degrees of freedom (right) in order to derive high-order approximations.

If we now zoom onto the right panel of Fig. 1 we could identify the following basis functions associated with each nodal value as shown in Fig. 2 where the panels show basis functions associated with the corner (left panel), edge (center panel) and interior (right panel) points of the element.



a) b) c)

Figure 2: Basis functions inside a 3rd order element. The left panel shows a corner, center panel shows an edge, and right panel shows an interior basis function.

Using such approximation functions one can then represent the state vector q as follows

$$q_N(x,t) = \sum_{i=1}^{M_N} \psi_i(x)\widetilde{q}_i(t) \tag{2}$$

where ψ are the basis functions defined in Fig. 2 and $\widetilde{q}$ represents expansion coefficients (the unknowns). The subscript N is used to denote that we are constructing an Nth degree approximation to the state vector q. The constant M_N defines the number of basis functions required to build an Nth degree approximation. For example, in one-dimension $M_N = N+1$ and on quadrilateral elements we would require $M_N = (N+1)^2$ while on triangles we would only require $M_N = \frac{1}{2}(N+1)(N+2)$. It is worth noting that the unified CGDG formulation is generally applicable to elements comprised of lines (1D), quadrilaterals and triangles (2D), hexahedra, tetrahedra, and triangular prisms (3D); however, we shall only describe the derivation for quadrilateral elements (2D).

2.2 Basis Functions, Differentiation, and Integration

To define the local operators which shall be used to construct the global approximation of the solution we began by decomposing the domain Ω into N_e non-overlapping quadrilateral elements such that

$$\Omega = \bigcup_{e=1}^{N_e} \Omega_e$$

as depicted in Fig. 1. We now define the reference element $\mathsf{I} = [-1,1]^2$ and introduce for each element Ω_e the smooth, bijective transformation $\mathscr{F}_{\Omega_e}$ such that $\Omega_e = \mathscr{F}_{\Omega_e}(\mathsf{I})$. The notation $x = \mathscr{F}_{\Omega_e}(\xi)$ will also be used, with $x = (x,z) \in \Omega_e$ and $\xi = (\xi,\eta) \in \mathsf{I}$. Associated with the map $\mathscr{F}_{\Omega_e}$ is the Jacobian $J_{\Omega_e} = \frac{d\mathscr{F}_{\Omega_e}}{d\xi}$, with determinant $|J_{\Omega_e}|$. The Jacobian determinant of $\mathscr{F}_{\Omega_e}$ restricted to the boundary of I is denoted by $J^s_{\Omega_e}$. In both the CG and DG methods, we require the construction of basis functions ψ that are functions of the space $\mathscr{P}_N(\mathsf{I})$ of bivariate polynomials of degree lower than or equal to N in I

$$\mathscr{P}_N(\mathsf{I}) = \text{span}\{\xi^n \eta^m \mid m,n \le N, (\xi,\eta) \in \mathsf{I}\}, \quad N \ge 1.$$

For a description of possible basis functions see, e.g., [6] for quadrilateral basis functions, [4] for triangular basis functions, and [9] for hexahedral basis functions.

With these definitions, we have

$$q_N(x,t)|_{\Omega_e} = \sum_{k=1}^{M_N} \psi_k\left(\mathscr{F}^{-1}_{\Omega_e}(x)\right) q_k(t), \qquad e = 1,\ldots,N_e, \tag{3}$$

where we introduce the grid points $x_k = \mathscr{F}_{\Omega_e}((\xi_k,\eta_k))$ and the grid point values $q_k(t) = q_N(x_k,t)$. The expansion given in Eq. (3) is essential in order to compute local element-wise derivatives and integrals. Concerning derivatives, in fact, it immediately provides

$$\left.\frac{\partial q_N}{\partial x}(x,t)\right|_{\Omega_e} = \sum_{k=1}^{M_N} \frac{d}{dx}\left[\psi_k\left(\mathscr{F}^{-1}_{\Omega_e}(x)\right)\right] q_k(t), \qquad \left.\frac{\partial q_N}{\partial t}(x,t)\right|_{\Omega_e} = \sum_{k=1}^{M_N} \psi_k\left(\mathscr{F}^{-1}_{\Omega_e}(x)\right) \frac{dq_k}{dt}(t). \tag{4}$$

Concerning the computations of integrals, the expansion defined by Eq. (3) yields

$$\int_{\Omega_e} q_N(x,t)\,dx = \int_{\mathsf{I}} q_N(x(\xi),t) \mid J_{\Omega_e}(\xi) \mid d\xi \simeq \sum_{k=1}^{M_Q} \omega_k\, q_k(t) \mid J_{\Omega_e}(\xi_k,\eta_k) \mid \tag{5}$$

where M_Q denotes the number of quadrature points of order Q.

2.3 CGDG Approximation

Now that we have defined the basis functions, differentiation, and integration we can begin the approximation of Eq. (1) via a unified CGDG formulation. We begin by substituting the approximations of q into the continuous partial differential equation. Then we multiply this approximation by a test function and integrate within each local element-wise domain and define the problem as follows: find $q_N(\cdot,t) \in V_N$ such that

$$\int_{\Omega_e} \psi\left(\frac{\partial q^e_N}{\partial t} + \nabla \cdot F(q^e_N)\right) d\Omega = \int_{\Omega_e} \psi\, S(q^e_N)\, d\Omega_e, \tag{6}$$

where q^e_N denotes the degrees of freedom collocated in Ω_e. Applying integration by parts and introducing the numerical flux F^*, the following problem is obtained:
find $q_N(\cdot,t) \in V_N$ such that $\forall \Omega_e$, $e = 1,\ldots,N_e$

$$\int_{\Omega_e} \psi \frac{\partial q^e_N}{\partial t} d\Omega_e + \int_{\Gamma_e} \psi\, n \cdot F^*(q_N)\, d\Gamma_e - \int_{\Omega_e} \nabla\psi \cdot F(q^e_N)\, d\Omega_e = \int_{\Omega_e} \psi\, S(q^e_N)\, d\Omega, \qquad \forall \psi \in V_N, \tag{7}$$

where

$$V_N = \{\psi \in \mathscr{S} : \psi \in \mathscr{P}_N(\mathrm{I}), e = 1,\ldots,N_e\}, \tag{8}$$

where for CG $\mathscr{S} = H^1(\Omega)$ whereas for DG it is $\mathscr{S} = L^2(\Omega)$.

Notice that the requirement $\psi \in H^1(\Omega)$ implies $V_N^{CG} \subset C^0(\Omega)$ whereas for V^{DG} this is not a requirement and so the solution may remain discontinuous across element boundaries.

By virtue of Eqs. (4) and (5), Eq. (7) can be written in the **weak form**

$$M^e \frac{dq^e}{dt} + \sum_{s=1}^{N_s} \left(M^{s,e}\right)^{\mathscr{T}} F^*(q) - \left(\widetilde{D}^e\right)^{\mathscr{T}} F(q^e) = S(q^e) \tag{9}$$

where

$$M^e_{hk} = \int_{\Omega_e} \psi_i \psi_j d\Omega_e, \qquad M^{s,e}_{hk} = \int_{\Gamma_e} \psi_i \psi_j n^{s,e} d\Gamma_e, \qquad \widetilde{D}^e_{hk} = \int_{\Omega_e} \nabla\psi_i \psi_j d\Omega_e,$$

where $h,k = 1,\ldots,M_N$, δ_{hk} is the Kronecker delta, $\xi_{k_{ij}} = (\xi_i, \eta_j)$, $w_{k_{ij}} = \omega_i\omega_j$, $w^s_{k_{ij}} = \omega_i$ for $j = 0$ or $j = N$ and $w^s_{k_{ij}} = \omega_j$ for $i = 0$ or $i = N$ where $k_{ij} = (i+1) + (N+1) * j$ denotes that the counter k is a tensor product of $i, j = 0,\ldots,N$.

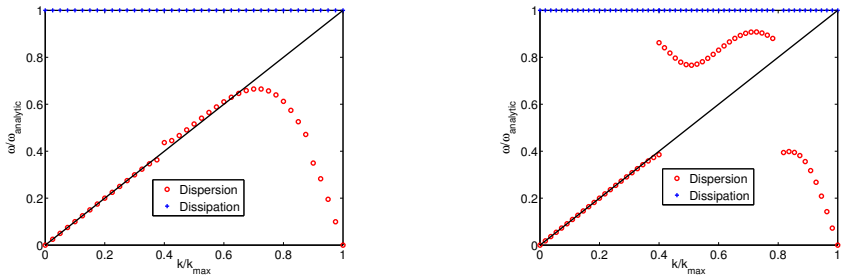


Figure 3: Stability Analysis with no dissipation for a) CG and b) DG using 4th order polynomials for the 1D wave equation.

Further integration by parts in Eq. (9), yields the following **strong form**

$$M^e \frac{dq^e}{dt} + \sum_{s=1}^{N_s} \left(M^{s,e}\right)^{\mathscr{T}} \left(F^*(q) - F(q^e)\right) + \left(D^e\right)^{\mathscr{T}} F(q^e) = S(q^e) \tag{10}$$

where

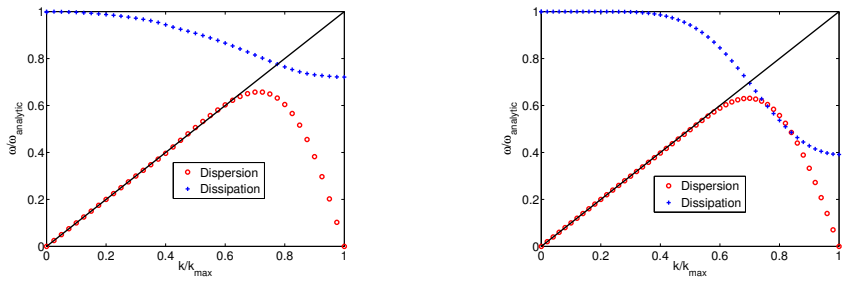
$$D^e_{hk} = \int_{\Omega_e} \psi_i \nabla\psi_j d\Omega_e.$$

Equations (9) and (10) define the DG solution as they are written. However, to make this form applicable to CG we have to add one important step known as the *Direct Stiffness Summation*, or DSS, and sometimes called the global assembly in the finite element literature. The idea is that we now construct global representations of the local element-wise problems given by Eqs. (9) and (10). Let us now define the following global matrices

$$M = \bigwedge_{e=1}^{N_e} M^e, \quad M^s = \bigwedge_{e=1}^{N_e} M^{s,e}, \quad D = \bigwedge_{e=1}^{N_e} D^e, \quad \widetilde{D} = \bigwedge_{e=1}^{N_e} \widetilde{D}^e$$

where $\bigwedge_{e=1}^{N_e}$ represents the mapping of the local degrees of freedom of each element Ω_e to the corresponding global degrees of freedom in Ω followed by summation (see [6] for further details).

Let us now look at the dispersion properties of both CG and DG. Figure 3 shows the amplification factors (blue crosses) and dispersion (red circles) for both the CG and DG methods without any dissipative mechanism. Of course, this setup is completely artificial since neither method is ever run without dissipation. However, it is interesting to see that the dispersion curves (red circles) are quite different from the theoretically expected curve (solid black line). By introducing some form of dissipation, we now get the curves given in Fig. 4.



a) b)

Figure 4: Stability Analysis with dissipation for a) CG with $\nu = 0.005$ and b) DG with upwinding using 4th order polynomials for the 1D wave equation.

We can now see that dissipation is indeed present (blue crosses) and this has significantly improved the dispersion relations for both CG and DG (red circles). The stationary waves at the far right are inconsequential since these waves are unresolved by the numerical method. Furthermore, the negative group velocities resulting from the negative slope of the dispersion relation beyond $k/k_{max} = 0.75$ are dampened and should be no cause for concern. A study for the linearized shallow water equations indeed show excellent dispersion properties for DG [1].

Like all high-order methods (with inherently little or no dissipation) element-based Galerkin methods such as the CGDG methods are not naturally monotone nor are they positivity preserving. These methods need some additional mechanisms in order to preserve positivity which we now discuss.

3 Positivity Preservation

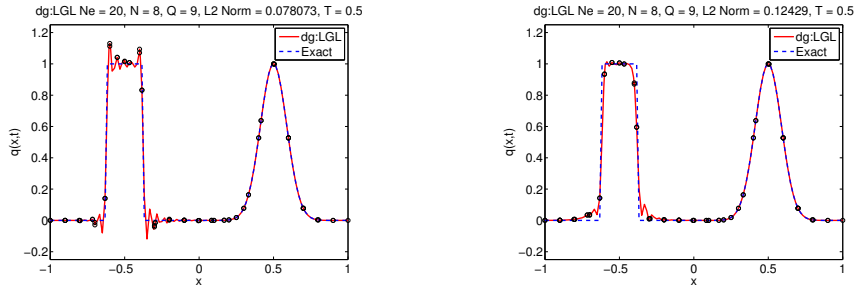
In DG methods the obvious choice for preserving positivity is the inclusion of limiters. In fact, much of the limiter technology built for finite volume methods in the early 1980s and 1990s could be used here. The main problem with those limiters (such as classical minmod limiters, etc) is that they were designed for low-order methods (e.g., second order finite volume methods). Since DG methods can be viewed as the high-order generalization of finite volume methods, then if we have chosen to use DG it is because we seek to use higher order than, say, second. Therefore, this situation demands the construction of a new set of limiters that: 1) maintain positivity and 2) does not destroy the high-order accuracy of the DG method.

Zhang and Shu [18] proposed the following linear scaling limiter for the compressible Euler equations

$$\widetilde{q}_i^e = \theta\left(q_i^e - \bar{q}^e\right) + \bar{q}^e, \qquad \theta = min\left\{\frac{\bar{q}^e - \varepsilon}{\bar{q}^e - q_{min}}, 1\right\} \tag{11}$$

where q_i^e is the DG solution inside the element e and on the quadrature point i, $\bar{q}^e$ is the mean value of q inside the element e, $q_{min} = min(q_i^e) \forall i = 0, \ldots, Q$, Q represents the order of the quadrature formula, and ε is a small number (e.g., 10^{-13}). This limiter is quite simple and rather effective. From Eq. (11) one can readily see that if $0 < q_{min} < \bar{q}^e$ that $\theta = 1$ and so $\widetilde{q}_i^e = q_i^e$. On the other hand, if $q_{min} < 0 < \bar{q}^e$, then $\theta = \frac{\bar{q}^e + \varepsilon}{\bar{q}^e - q_{min}}$ which then means that $\widetilde{q}_i^e = \varepsilon > 0$ when i corresponds to the minimum point inside of e. Clearly, this limiter maintains positivity as long as $\bar{q}^e$ is positive. This assumption is not too restrictive because we assume (e.g., for moisture) that the variable is initially positive everywhere and so the mean value inside each element should always be positive. Once we integrate the equations forward in time as long as we have a positivity preserving limiter in space and use a strong stability preserving method in time then $\bar{q}^e$ should always remain positive. Moreover, it is easy to see from Eq. (11) that this limiter preserves conservation. This can be shown straightforwardly by recognizing that q_i^e can be written in modal form as $q_i^e = \bar{q}^e + H.O.T$ where the high-order terms (H.O.T) are expanded via orthogonal polynomials. Since $P_0 = 1$ is the first orthogonal (Legendre) polynomial, then the integral of the remaining H.O.T terms, multiplied by $P_0 = 1$, will vanish.

What is not obvious to see (and of great importance) is that the limiter given in Eq. (11) also retains high-order convergence. In other words, if we choose to use high-order methods, the solution remains high-order. For a proof of the preservation of high-order convergence see [18]. Let us now demonstrate how this limiter works in practice.

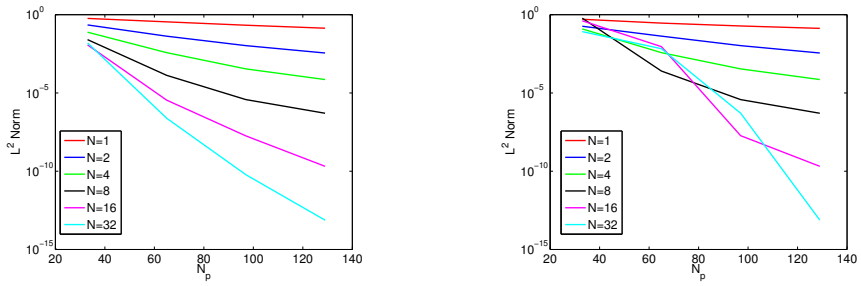


a) b)

Figure 5: Performance of DG Positivity Preserving Limiter. The 9th order DG solution after one-half revolution is shown for a) no limiter and b) the positivity preserving limiter.

Let us solve the one-dimensional wave equation: $\frac{\partial q}{\partial t} + \frac{\partial}{\partial x}(qu) = 0$ where $u = constant$ with periodic boundary conditions. Figure 5 shows the snapshot of the solution after a half revolution around the periodic domain using 8th degree polynomials (9th order accurate). Note that in Fig. 5a we see that the high-order DG method has no problem resolving the smooth part of the wave but has major issues with maintaining positivity with the non-smooth square wave. When we apply the limiter (Fig. 5b) we see that the method now maintains positivity. Note however, that minor overshoots are still present in the solution. It is possible to eliminate these overshoots by a similar limiter based on a strict maximum principle but this assumes that we know what the maximum value can be (see [17]). To see how the limiter affects the high-order accuracy of the DG method, we now run a convergence study for various polynomial orders after one full revolution but only considering the smooth part of the curve (we do not expect high-order convergence for the non-smooth curve since it is not infinitely differentiable). Figure 6a shows the convergence rates with no limiter while Fig. 6b shows the convergence rates with the limiter. Although the curves are different, the general high-order traits are retained. This convergence rate study may not seem so interesting at first glance but in order to perform it we ran each polynomial order through a variety of different numbers of elements. For the under-resolved configurations the limiter was automatically activated in order to combat any negative numbers no matter how small. For these simulatons we took $\varepsilon = 0$ so we do indeed get perfect positivity for all the simulations shown in this

figure. For CG methods, this limiter can be applied provided that the CG method is constructed in the unified CGDG formulation described above; the results are similar to those shown for the DG method (not shown). However, one main drawback of the CG method is that it is inherently non-dissipative. The DG method has natural dissipation built into it by virtue of the numerical flux (e.g., approximate Riemann solver) where, if upwinding is used, yields sufficient dissipation to combat most steep gradients (see [6] for examples of CG versus DG for strong gradients). Therefore, the CG method needs a dissipative mechanism in addition to the positivity preserving limiter. One possibility for handling this stabilization (and perhaps the positivity preservation as well) is to use Variational Multi-Scale (VMS) methods [8]. The idea behind VMS is very much akin to SUPG (Streamwise-Upwind Petrov-Galerkin) methods where a stabilization (diffusion) term is added along the direction of the fluid flow. However, VMS extends this idea by using not just the advective terms but the entire operator. Let us now describe the VMS method applied to high-order CG methods. This description follows closely the approach of [12].



a) b)

Figure 6: Convergence of DG Positivity Preserving Limiter. The convergence rates are shown for a smooth (Gaussian) problem with a) no limiter and b) with the positivity preserving limiter.

Let us write the system of conservation laws as follows:

$$R \equiv \frac{\partial q}{\partial t} + \mathscr{L}(q) - S(q) = 0 \tag{12}$$

where $\mathscr{L}$ is the, e.g., divergence of the flux tensor for system of hyperbolic equations. In short hand notation, the discretization of Eq. (12) by the standard CG method is written as

$$\left(\psi, \frac{dq_N}{dt}\right) + (\psi, \mathscr{L}(q_N)) = (\psi, S(q_N)) \tag{13}$$

where $(u,v) = \bigwedge_{e=1}^{N_e} \int_{\Omega_e} uv d\Omega_e$ denotes that the DSS operator has been applied. Following the steps in [12] we arrive at the VMS formulation of Eq. (13) as

$$\left(\psi, \frac{dq_N}{dt}\right) + (\psi, \mathscr{L}(q_N)) = (\psi, S(q_N)) + (R, \tau \mathscr{L}^*(\psi)) \tag{14}$$

where τ is the stabilization parameter that has been analyzed extensively by the SUPG and VMS community. In [12] we derive a simple parameter for use with high-order CG methods for the advection-diffusion equation as follows:

$$\tau = \frac{h}{2u}\left(\coth(Pe^e) - \frac{1}{Pe^e}\right) \tag{15}$$

where h is the element length, u is the velocity, and Pe^e is the element Peclet number defined as $Pe^e = \frac{uh}{2\nu}$ where ν is the kinematic viscosity of the problem. The attraction of the VMS method is that the

dissipation (last term in Eq. (14)) acts only when the residual is non-zero. When the method is able to satisfy the governing PDE this term vanishes.

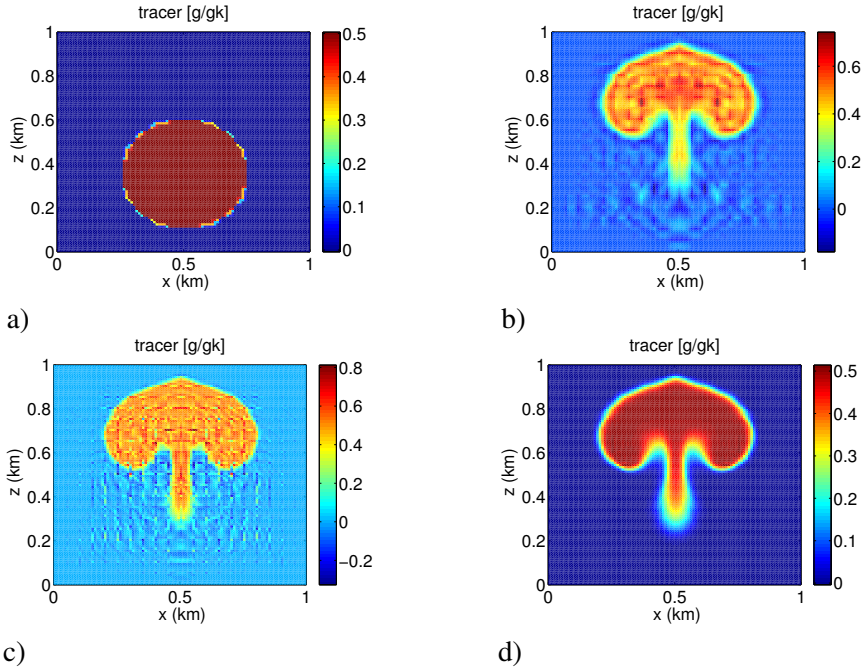


Figure 7: Deformation of a tracer imbedded inside of a rising thermal bubble for the compressible Euler equations using 4th order CG: for a) the initial condition of the tracer, b) solution with filtering, c) solution with hyper-viscosity ($\nu = 0.001 m^2/s$), and d) VMS with Discontinuity Capturing. All the results are for a 20×20 element grid with 4th order polynomials up to a 600 second integration.

Figure 7 shows the results for a discontinuous tracer with various stabilization methods for CG. Figure 7a shows the initial condition, while Figs. 7b, c, d show the results after 600 seconds. Figure 7b shows the results using standard filters for CG methods, Fig. 7c uses artificial diffusion, and Fig. 7d uses VMS with Discontinuity Capturing (see [12] for details). Clearly, the VMS result is by far the best and is able to get the tracer perfectly.

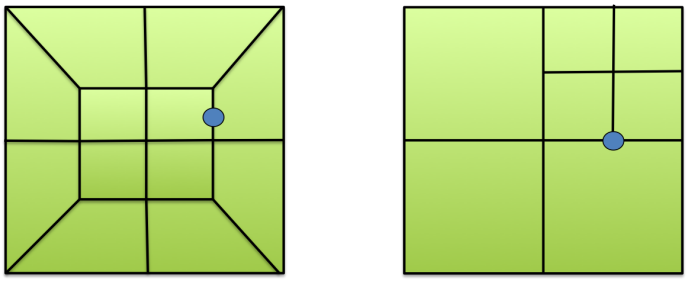
4 The NUMA Model

The Nonhydrostatic Unified Model of the Atmosphere (NUMA) is a framework for developing unified regional and global weather prediction models. NUMA contains unified CGDG methods as well as a suite of explicit, IMEX, and fully-implicit time-integrators (see [3] for some possible time-integrators). In addition, NUMA has the capability to handle adaptive mesh refinement for both the CG and DG implementations (see Sec. 5). Finally, NUMA has a collection of modules that allow it to simulate: 1) nonhydrostatic equations in 2D; 2) nonhydrostatic equations in 3D (both in a cube and on the globe, see [9, 3] for examples); 3) shallow water equations on the plane with wetting and drying for both storm-surge and tsunami simulations; and 4) shallow water equations on the sphere. NUMA-Atmosphere is being targeted by the U.S. Naval Research Laboratory in Monterey, California as its next-generation mesoscale (regional) and global model. The development of the NUMA model is based on a synergistic collaboration between the Naval Postgraduate School (NPS) and the Naval Research Laboratory (NRL) - Monterey. The effort at NPS primarily consists of mathematical studies, algorithm development involving primarily the dynamics and simple moist physics, while the effort at NRL focuses mainly on preparing NUMA for operations; this includes the inclusion of topography, databases, radiation, and other necessary physical parameterizations.

5 Adaptive Mesh Refinement

There are two classes of Adaptive Mesh Refinement (AMR): 1) conforming and 2) non-conforming.

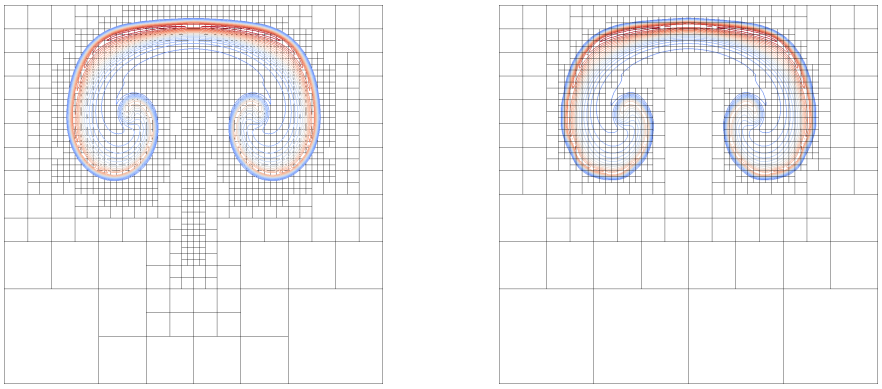
Figure 8a shows a conforming grid while Fig. 8b shows a non-conforming grid. The difference is that the blue dot on the left panel shows that this edge is only claimed by two quadrilateral elements (conforming) whereas on the right panel the edge with the blue dot shows that the edge is claimed by 3 elements (2 smaller ones from above and one larger one below). Conforming AMR is quite standard for triangles but difficult to do for quadrilaterals although not impossible. Conforming AMR places all the burden on the grid generation software to generate conforming grids whereas the CGDG solver requires no modification in order to handle this type of grid - these are the virtues of this approach. The vices is that conforming AMR software is typically very sophisticated and so may require coupling the solver to an external grid generation library (see, e.g., [13]).



a) b)

Figure 8: Two types of adaptive mesh refinement: a) conforming and b) non-conforming.

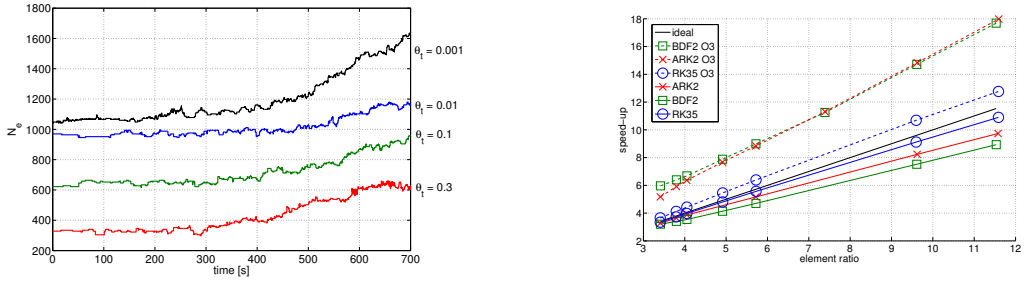
On the other hand, Fig. 8b shows an example of non-conforming AMR which at first glance seems much more complex. The advantage of this approach is that the grid generation can be made to be quite simple. For example, whenever the refinement criterion indicates the need for refinement, each quadrilateral is divided into four smaller (children) elements. Non-conforming AMR, in contrast to conforming AMR, places all the burden on the CGDG solver because the solver now has to understand how to handle non-conforming edges. For details of AMR, we refer the reader to our recent paper [11].



a) b)

Figure 9: Potential temperature contours and mesh for a rising thermal bubble after 700 seconds with a) $\theta_t = 0.001$ and b) $\theta_t = 0.05$ where θ_t is the refinement criterion threshold value.

Figure 9 shows the potential temperature contours and grid for two different refinement criteria (θ_t). The smaller θ_t means that more elements are produced since once that value of θ is detected, the mesh is refined. One obvious question is: what is there to be gained by using dynamic AMR? This question is answered in detail in [11] but in Fig. 10 we show an example.



a) b)

Figure 10: The benefit of AMR is illustrated. In a) we show the time history of the change in the number of elements with different refinement criteria θ_t and in b) we show the speedup over a high-resolution uniform mesh for different time-integrators (explicit and implicit-explicit methods)

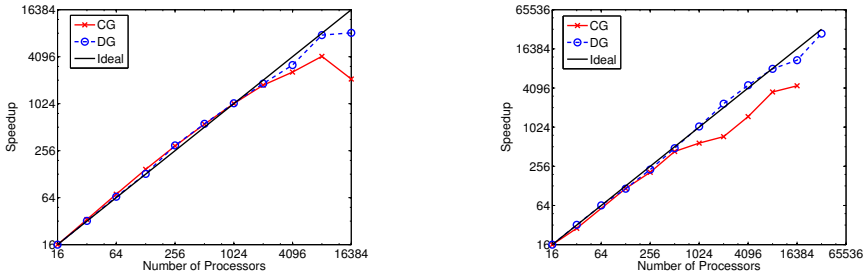
.

Figure 10a shows the number of elements as a function of time using four different refinement criteria (θ_t values). Clearly, the larger θ_t the fewer elements produced. In Fig. 10b we show the speed-up gained by using AMR. The speed-up is measured as the ratio of the wallclock time required by a high-resolution reference simulation to the AMR simulation. The ideal speed-up assumes that the wallclock time is directly correlated to the number of elements in the domain. This figure shows that ideal speed-up is indeed achieved by the explicit RK35 method (solid blue line) when no compiler optimization is used. In contrast, the IMEX time-integrators (solid red and green lines) fall short of ideal speed-up but it should be understood that these simulations are well over a factor of ten faster than the explicit simulation (by virtue of the larger time-step allowed). The dotted lines in this figure show the results when compiler optimizations are turned on. Here we note that the IMEX methods achieve super speed-up. The reasons are due to the AMR simulations having fewer elements that fit entirely within cache and with the compilers streamlining memory access (e.g., prefetching, etc.). We clearly see a gain in efficiency with AMR but this efficiency does come at a price regarding accuracy (not shown). Therefore AMR is only a choice when one is interested in resolving a particular process accurately where one is willing to pay the price of inaccuracies in the coarser mesh regions. One such example is the tracking of a hurricane where the fine region is focused on the eye of the storm whereas the rest of the domain remains coarse.

6 Parallelization

One of the main virtues of CGDG methods is their excellent scalability on distributed-memory architectures. The reason for this has to do with the building-block approach of element-based Galerkin (EBG) methods. Although the finite volume (FV) method can be classified as an EBG method, it is the high-order extension of the FV method that makes it no longer fall within the EBG category. For a method to satisfy the EBG property it needs to have a compact stencil - this means that all (or most of) the information required to build an approximation falls within the building-block (or element). This is true for both CG and DG methods because each element can contribute to the overall solution independently. This is not the case if the differencing stencil is non-compact because information from other building-blocks

(or control volumes) is required. This adversely affects parallel scalability as the amount of information being transferred needs to be much larger than for a compact method in spite of the main cost in parallel algorithms being latency. In Fig. 11 we show the scalability of CG and DG methods in the NUMA model.

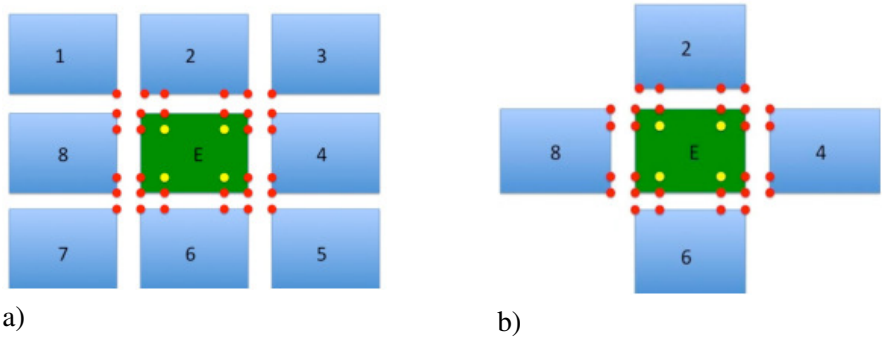


a) b)

Figure 11: The scalability of CGDG methods in NUMA using explicit time-integration for 32^3 hexahedral elements. In a) we show the results for 4th order polynomials (5^3 grid points per element) and in b) are the results for 8th order polynomials (9^3 grid points per element).

The left panel (Fig. 11a) shows the speed-up using 4th order polynomials inside each of the 32^3 cubed elements in this 3D calculation of a rising thermal bubble. Although both CG and DG scale very well, they begin to lose perfect scalability at 16K cores. In Fig. 11b we show the results when we increase the polynomial order to 8th order which dramatically increases the amount of work being done (on-processor) inside each element. The DG method achieves perfect strong linear scalability beyond 30K cores; at this point there is only one DG element per core. In contrast, the CG method continues to scale well although not perfectly. When this simulation was conducted, the CG and DG methods resided in different codes. Now that they reside within the same framework we believe that the CG result will approach those of DG although will not yield the same level of efficiency.

To understand why DG scales better than CG we need to understand the communication stencil of both methods.



a) b)

Figure 12: The communication stencil for a) CG and b) DG on quadrilateral elements.

Figure 12 shows the communication stencils for both CG and DG. Since the solution for CG is required to be C^0 continuous then all degrees of freedom that intersect at an element edge or corner must have the same solution regardless of which element they are defined in; this is depicted in Fig. 12a where we see that the center element requires information (and hence communication) from all of the eight surrounding elements which we refer to as *vertex neighbors* since these neighbors share a vertex with the

center element. In contrast, the DG method (Fig. 12b) only requires information from the 4 surrounding edge neighbors and so the communication stencil is much smaller. However, it should be understood that we have made a conscious decision to construct the communication stencil in this fashion (via the numerical flux function F^*) which then limits the maximum allowable time-step compared to the CG method. We could construct a different (multi-dimensional) flux (see, e.g., [16]) that would then require a communication stencil similar to CG. The advantage is that the time-step would be similar to that for CG but the disadvantage is that the communication stencil is larger. We believe that the more compact communication stencil outweighs the time-step restriction especially for very large core counts as those expected in the petascale and exascale ranges.

It is generally understood that to reach the exascale range of computing (10^{18} floating point operations per second) will require a hybrid computer system consisting of both Central Processing Units (CPUs) and Graphics Processing Units (GPUs). To take advantage of such a computer system will require rewriting existing message-passing interface (MPI) code to allow the use of either CUDA or OpenCL or some accelerator translator (GPUs are often called accelerators). There is an on-going U. S. Office of Naval Research (ONR) project related to porting the NUMA model to GPUs by writing general modules (using OpenCL) called "mini-apps" that are highly optimized for specific accelerator hardware and then ported into specific models. The translation from CPU-based to GPU-based kernels will be handled by Loo.py [10], a newly developed translator that takes a specific syntax and produces highly optimized kernels; the advantage of Loo.py is that it offers a semi-automated process of producing highly optimized kernels on future accelerator architectures. This ONR project consists of a collaborative effort involving the following institutions: the Naval Postgraduate School, (Department of Applied Mathematics), Rice University (Computational and Applied Mathematics), University of Illinois-Urbana Champaign (Computer Science), and the Naval Research Laboratory (both in Monterey California and Stennis Mississippi). Software developed in this project will be open source so that both the atmospheric and ocean modeling communities can benefit from this work.

7 Conclusions

We have shown a unified continuous Galerkin (CG) and discontinuous Galerkin (DG) formulation for systems of nonlinear hyperbolic equations. Furthermore, we have shown some possible positivity preserving mechanisms for both CG and DG. The dispersion relations of CGDG were presented and the parallel scalability of these methods was illusrated for core counts beyond 30K cores. Finally, the use of AMR for CGDG was illustrated. In summary, CGDG methods appear to be a very good choice for the construction of next-generation nonhydrostatic unified regional/global numerical weather systems. CGDG methods are: highly accurate, efficient, conservative with excellent dispersion properties (at the resolvable scales), geometrically flexible to allow the use of statically and dynamically adaptive (non-conforming) mesh generation, highly scalable on current and future computer architectures, and sufficiently flexible to allow models based on these methods to have a long shelf-life.

Acknowledgements

The author gratefully acknowledges the support of the Office of Naval Research through program element PE-0602435N, the National Science Foundation (Division of Mathematical Sciences) through program element 121670, and the Air Force Office of Scientific Research through the Computational Mathematics program. The author also gratefully acknowledges the support of the European Center for Medium-Range Weather Forecasting. Finally, the author would like to thank Simone Marras and Michal Kopera for comments on the manuscript.

References

[1] P. E. BERNARD, E. DELEERSNIJDER, V. LEGAT, AND J. F. REMACLE, *Dispersion analysis of discontinuous Galerkin schemes applied to Poincare, Kelvin and Rossby waves*, Journal of Scientific Computing, 34 (2008), pp. 26–47.

[2] A. FOURNIER, M. TAYLOR, AND J. TRIBBIA, *The spectral element atmosphere model (SEAM): High-resolution parallel computation and localized resolution of regional dynamics*, Monthly Weather Review, 132 (2004), pp. 726–748.

[3] F. GIRALDO, J. KELLY, AND E. CONSTANTINESCU, *Implicit-explicit formulations for a 3d nonhydrostatic unified model of the atmosphere (numa)*. in press SIAM Journal on Scientific Computing, 2013. http://faculty.nps.edu/fxgirald/Homepage/Publications_files/Giraldo_et_al_SISC_2013_Final.pdf.

[4] F. X. GIRALDO, *High-order triangle-based discontinuous Galerkin methods for hyperbolic equations on a rotating sphere*, J. Comp. Phys., 214 (2006), pp. 447–465.

[5] F. X. GIRALDO, J. S. HESTHAVEN, AND T. WARBURTON, *Nodal high-order discontinuous Galerkin methods for the spherical shallow water equations*, J. Comp. Phys., 181 (2002), pp. 499–525.

[6] F. X. GIRALDO AND M. RESTELLI, *A study of spectral element and discontinuous Galerkin methods for the Navier–Stokes equations in nonhydrostatic mesoscale atmospheric modeling: Equation sets and test cases*, J. Comp. Phys., 227 (2008), pp. 3849–3877.

[7] F. X. GIRALDO AND T. E. ROSMOND, *A scalable spectral element eulerian atmospheric model (SEE-AM) for NWP: Dynamical core tests*, Monthly Weather Review, 132 (2004), pp. 133–153.

[8] T. HUGHES, G. FEIJOO, L. MAZZEI, AND J. QUINCY, *The variational multiscale method - a paradigm for computational mechanics*, Computer Methods in Applied Mechanics and Engineering, 166 (1998), pp. 3–24.

[9] J. F. KELLY AND F. X. GIRALDO, *Continuous and discontinuous Galerkin methods for a scalable three-dimensional nonhydrostatic atmospheric model: Limited-area mode*, Journal of Computational Physics, 231 (2012), pp. 7988–8008.

[10] A. KLÖCKNER, *Loo.py: A loop generation tool for CPUs and GPUs*. Rice 2012 Oil & Gas High Performance Computing Workshop Lecture, 2012. http://oghpc.blogs.rice.edu/files/2011/09/Kloeckner-final.pdf.

[11] M. KOPERA AND F. X. GIRALDO, *Analysis of adaptive mesh refinement for imex discontinuous galerkin solutions of the compressible euler equations in atmospheric simulations*. submitted to the Journal of Computational Physics, 2013. http://faculty.nps.edu/fxgirald/Homepage/Publications_files/Kopera_Giraldo_JCP_2013_submitted.pdf.

[12] S. MARRAS, J. F. KELLY, F. X. GIRALDO, AND M. VAZQUEZ, *Variational multiscale stabilization of high-order spectral elements for the advection-diffusion equation*, Journal of Computational Physics, 231 (2012), pp. 7187–7213.

[13] A. MÜLLER, J. BEHRENS, F. X. GIRALDO, AND V. WIRTH, *Comparison between adaptive and uniform discontinuous Galerkin simulations in dry 2D bubble experiments*, Journal of Computational Physics, 235 (2013), pp. 371–393.

[14] R. NAIR, H.-W. CHOI, AND H. TUFO, *Computational aspects of a scalable high-order discontinuous galerkin atmospheric dynamical core*, Comp. Fl., 38 (2009), pp. 309–319.

[15] M. TAYLOR, J. TRIBBIA, AND M. ISKANDARANI, *The spectral element method for the shallow water equations on the sphere*, Journal of Computational Physics, 130 (1997), pp. 92–108.

[16] L. YELASH, A. MÜLLER, M. LUKACOVA-MEDVIDOVA, F. X. GIRALDO, AND V. WIRTH, *Adaptive discontinuous evolution galerkin method for dry atmospheric flow*, J. Comp. Phys., (in review 2013).

[17] X. ZHANG AND C.-W. SHU, *On maximum-principle-satisfying high order schemes for scalar conservation laws*, Journal of Computational Physics, 229 (2010), pp. 3091–3120.

[18] ——, *On positivity-preserving high order discontinuous Galerkin schemes for compressible Euler equations on rectangular meshes*, Journal of Computational Physics, 229 (2010), pp. 8918–8934.