QUADRATURE-FREE IMPLEMENTATION OF THE DISCONTINUOUS GALERKIN METHOD FOR HYPERBOLIC EQUATIONS

H. L. Atkins*
NASA Langley Research Center, Hampton, VA 23681

Chi-Wang Shu†
Brown University, Providence, RI 02912

A discontinuous Galerkin formulation that avoids the use of discrete quadrature formulas is described and applied to linear and nonlinear test problems in one and two space dimensions. This approach requires less computational time and storage than conventional implementations but preserves the compactness and robustness inherent to the discontinuous Galerkin method. Test problems include both linear and nonlinear one-dimensional scalar advection of both smooth and discontinuous initial value problems, two-dimensional scalar advection of smooth initial value problems discretized by using unstructured grids with varying degrees of smoothness and regularity, and two-dimensional linear Euler solutions on unstructured grids.

Introduction

Computational methods for aeroacoustics must possess accuracy properties that exceed those of conventional second-order computational fluid dynamics (CFD) methods. At the same time, many problems of interest involve complex geometries that are not easily treated by common high-order methods that usually require a smooth, structured grid. In addition to the geometrically complex problem, we are particularly interested in strongly nonlinear flows that contain shock waves as a major source of sound generation, such as in the case of jet noise.

In an effort to satisfy these requirements, the relatively untried discontinuous Galerkin (DG) method is being tested for hyperbolic problems. Some advantages of this approach include the ease with which the method can be applied to both structured and unstructured grids and its suitability for parallel computer architectures. The approach also has several useful mathematical properties.

Johnson and Pitkärata[1] proved stability and error estimates for linear scalar advection. In a series of papers, Cockburn, Shu, et al[2–4] discussed the DG method using approximate Riemann solvers, limiters, and total-variation diminishing (TVD) Runge-Kutta time discretizations for nonlinear hyperbolic problems. In reference 2, the general formulation in one space dimension is provided, with a detailed description and analysis of accuracy, stability (in terms of total variation), and implementation. Numerical examples of scalar equations are also provided. In reference 3, the method is applied to one-dimensional systems. Stability of the initial boundary value problem for a linear system is proved, and numerical experiments were performed for the one-dimensional Euler equations. In reference 4, the method is generalized to multispace dimensions. A main result in reference 4 is the design of a total-variation bounded (TVB) limiter that applies to general triangulations, maintains high order in smooth regions, and guarantees maximum norm stability. Jiang and Shu[5] have also proved that the DG method satisfies a local cell entropy inequality for the square entropy, for arbitrary triangulations in any space dimension, and for any order of accuracy. This trivially implies L_2 stability of the method for nonlinear shocked problems in the scalar case.

Although the DG method has not been widely used in the CFD arena, several instances exist in which this method has been applied to the Euler or Navier-Stokes equations.[3, 4, 6–9] Halt and Agarwal[6] applied the method of moments (similar to the DG method) to the steady two-dimensional Euler equations for subsonic flows. Bassi and Rebay[7] applied the DG method to two-dimensional Euler equations for transonic flows and demonstrated the importance of properly treating curved boundaries. In reference 8, Bassi and Rebay extended their method to the Navier-Stokes equations by introducing the gradient of the solution as an auxiliary variable. Most recently, Lowrie, Roe, and van Leer[9] presented a fully discrete DG method for the unsteady Euler equations. Biswas, Devine, and Flaherty[10] applied the DG method in an h-p version in the adaptive grid environment and considered the issues of limiters for moments, as well as for parallel implementations.

In the works described above, the integrals that appear in the formulation are evaluated with quadrature formulas. In this particular work, the DG method is implemented in a form that avoids the use of quadra-

* Research Scientist, Aerodynamic and Acoustic Methods Branch, Fluid Mechanics and Acoustics Division, Senior Member AIAA.

† Professor, Division of Applied Mathematics.

ture formulas. This implementation reduces both the storage requirements and the operation count. In the first section, the DG method is described in general. Then, additional motivation for using the quadrature-free approach is given, along with specific details of the formulation. Storage requirements and operation count are also discussed, and the results of a stability analysis are given. It is shown that the primary arguments against the method, high storage requirements and a high operation count, are unjustified. The last section presents numerical results for one- and two-dimensional test problems. A linear scalar equation is used to verify the general properties of the DG method for solution expansions up to 12th order. The nonlinear Burger's equation is used to demonstrate the shock-capturing capabilities of the method in one space dimension. The method is applied to both scalar advection and the linear Euler equations in two space dimensions to demonstrate the unstructured grid capability. This work focuses on the new formulation of the DG method and defers all discussion of boundary conditions to future articles. Hence, all test problems are treated as spatially periodic.

General Discontinuous Galerkin Method

Consider an arbitrary domain Ω in which the solution is governed by a conservation equation of the form

$$U_t \; + \; \nabla \cdot \vec{F} \; = \; 0 \tag{1}$$

The DG method can be arrived at by partitioning the domain onto smaller, nonoverlapping elements Ω_i that cover the domain and then applying a traditional Galerkin[11] method to each element. The Galerkin approach within an element is defined by selecting a finite-dimensional basis set, approximating the solution as an expansion in that basis, and then projecting the governing equation onto each member of that basis set.

$$B \; \equiv \; \{b_k, \; 0 \leq k \leq N(p,d)\} \tag{2}$$

$$U_{\Omega_i} \; \approx \; V_i \; \equiv \; \sum_{j=0}^{N} v_{i,j} \, b_j \tag{3}$$

$$\int_{\Omega_i} b_k \left(V_t \; + \; \nabla \cdot \vec{F} \right)_i d\Omega \; = \; 0 \qquad k = 0, 1, 2, \ldots, N \tag{4}$$

The basis set must be constructed from the lower order terms of a complete and linearly independent set. In the fully discrete approach,[9] the basis set contains both temporal and spatial functions. In the semi-discrete approach, which is used here, the basis set contains only spatial functions, and the solution expansion coefficients $v_{i,j}$ are functions of time. The number of terms in the expansion, $N(p,d)+1$, depends on the order (or degree) of the expansion p and the number of time-space dimensions d represented. To streamline the notation, $N(p,d)$ will be denoted simply by N except where the full form is needed for clarity.

The divergence term is recast by applying integration by parts as

$$\int_{\Omega_i} b_k \sum_{j=0}^{N} (v_{i,j})_t b_j \, d\Omega \; - \; \int_{\Omega_i} \nabla b_k \; \cdot \; \vec{F}_i d\Omega + \int_{\partial\Omega_i} b_k \, \vec{F}_i^R \; \cdot \; d\vec{s} \; = \; 0 \qquad k = 0, 1, 2, \ldots, N \tag{5}$$

where $d\vec{s}$ is an outward-pointing surface-element normal and the flux vector $\vec{F}^R$ is some approximation to a Riemann flux that depends on the solutions in both element i and the neighboring element.

Although equation (5) could be evaluated in physical space, the equation can be conveniently (and sometimes advantageous) represented in terms of coordinates that are local to the element. After such a mapping, (5) becomes

$$\int_{\Omega_i} b_k \sum_{j=0}^{N} (v_{i,j})_t b_j \; J_i d\Delta \; - \; \int_{\Omega_i} \nabla b_k \; \cdot \; \mathbf{J}_i^{-1} \vec{F}_i \; J_i d\Omega + \int_{\partial\Omega_i} b_k \, \mathbf{J}_i^{-1} \vec{F}_i^R \; \cdot \; J_i d\vec{s} \; = \; 0 \qquad k = 0, 1, 2, \ldots, N \tag{6}$$

where

$$\mathbf{J}_i \; \equiv \; \frac{\partial(x,y,z)}{\partial(\xi,\eta,\zeta)} \, , \qquad J_i \; = \; |\mathbf{J}_i|$$

and (ξ, η, ζ) are the local coordinates within the element.

Depending on the choice of basis functions and the form of J_i, the first integral can usually be simplified. If, for example, the basis set is orthonormal with respect to J_i, then the first integral term in (6) reduces to an identity mass matrix. In most implementations, either the transformation is an isoparametric form (thus, J_i is a polynomial) or the element is allowed to be an arbitrary polygon. In either case, the temporal term can be expressed as $\mathbf{M}_i \; \mathbf{V}_t$, where $\mathbf{M}_i \; \equiv [m_{k,j}]$ is the

mass matrix, $\mathbf{V} \equiv [v_l]$, and

$$m_{k,j} = \int_{\Omega_i} b_k \, b_j \, J_i \, d\Omega \tag{7}$$

In most cases, the mass matrix can be computed and inverted in advance of the main calculation; however, it must be stored for every element. In all previous implementations, the spatial integrals are evaluated with quadrature formulas that are appropriate to the element shape and the required degree of accuracy. The resulting $N+1$ equations are then used to evolve in time the coefficients of the solution expansion $v_{i,j}$.

In the present formulation, we place a fundamental restriction on the types of elements that are permitted. In particular, we require that all elements (except perhaps elements near a curved boundary) have a linear mapping to a simple similarity element, such as an equilateral triangle or a square in two dimensions (or similar simple similarity elements in three dimensions). With this restriction, the temporal term can be rewritten as $J_i \mathbf{M} \, \mathbf{V}_t$, where the mass matrix $\mathbf{M}$ is now defined by

$$m_{k,j} = \int_{\Omega_i} b_k \, b_j \, d\Omega \tag{8}$$

As a consequence, $\mathbf{M}$ is the same for all elements of a given similarity shape (e.g. one $\mathbf{M}$ is applicable to all triangles, another is applicable to all squares, etc.) except those near a curved boundary. A considerable reduction is realized in the storage requirements that far out weights any inconvenience that arises from the restriction in element shapes.

Throughout the remainder of this discussion, the subscript that identifies the element will be omitted unless its use is necessary for clarity. In addition, although the above discussion applies to a general basis set, the remaining discussion will assume the basis functions are polynomials. In particular, the results presented later all use a basis constructed from simple monomials (e.g. $B \equiv \{1, x, y, x^2, xy, y^2, \ldots\}$.)

Quadrature-Free Approach

Quadrature formulas, especially Gaussian quadrature, are usually the most accurate and efficient means of evaluating integrals. However, this feature is based on the assumption that the data are readily available at the quadrature points (i.e., the data are stored there), which is usually the case for most finite-element and spectral-element methods. In these methods, the unknown variables are the values of the solution at the quadrature points, and the basis or "shape" functions are often the Lagrangian polynomials associated with the quadrature points. In the DG method, however, the formulation requires the evaluation of both volume and surface integrals, and no single set of $N+1$ quadrature points exist that can be used to evaluate all the integrals to the required accuracy. Thus, the usual practice in the implementation of the DG method is to choose a basis and to store the expansion coefficients. As a consequence, the evaluation of the volume integral, for instance, requires $N+1$ operations at each quadrature point simply to obtain the data needed to evaluate the quadrature formula. Because all integrals must be exact for polynomials of degree $2\,p$, the operation count for the complete evaluation of the volume integral is on the order of $(N+1)^2$ operations for each equation, assuming optimal quadrature formulas exist.

In contrast, if the basis functions are judiciously chosen such that integrals of products of the basis functions can be evaluated exactly, then the complete volume integral can be evaluated exactly in only $N+1$ operations. A more precise comparison of the operation count is given in a later section. In the quadrature-free implementation, we derive a set of matrices that accomplish exactly this task. Further, as a consequence of the restriction previously placed on the type of elements, these matrices are the same for all elements of a given type (e.g. one set of matrices applies to all triangles, another set applies to all squares, etc.). The result is a low storage and low computational cost method that retains the ability of the DG method to treat unstructured grids in a accurate and robust manner.

Flux Expansion

To accomplish this task, however, an efficient method for expanding the flux vector $\vec{F}$ in terms of the basis set is needed.

$$\vec{F} = \sum_{j=0}^{M} \vec{f}_j \, b_j \qquad M \geq N(p,d) \tag{9}$$

where the number of terms in the expansion M depends on the form of the nonlinearity in $\vec{F}$.

The expansion is trivially accomplished when $\vec{F}$ is a linear function of U. Similarly, for common test problems such as the nonlinear Burger's equation, $\vec{F} = \frac{1}{2} U(x)^2$ can be obtained by multiplication as long as triple (and higher) products of the basis function are also easily integrated. Such is the case when the basis functions are polynomials in the local coordinates of a simple element.

The more complex flux functions, such as those of the Euler equations, can be treated in several ways. One approach is that of Lowrie et al[9] in which the

solution expansion is applied to the parameter vector $\left[\sqrt{\rho},\ \sqrt{\rho}\,u,\ \sqrt{\rho}\,v,\ \sqrt{\rho}\,H\right]$ instead of the conserved variables $[\rho,\ \rho u,\ \rho v,\ \rho e]$. Although this approach allows all nonlinear fluxes to be evaluated exactly, the coefficient matrices become dependent on the local solution. Hence the coefficient matrices derived here would be different for every element and would need to be recomputed at every time step; the storage requirements and computational effort would be unacceptably high.

A general alternative is to expand the flux in a Taylor series. Because the Euler equations consist of terms such as $(\rho u)^2/\rho$ and $(\rho u)(\rho v)/\rho$, a practical procedure is to approximate ρ^{-1} in the basis functions by using a Taylor series and then use multiplication to complete the flux expansion.

Another approach is to define the flux in terms of the projection operator as follows. Let

$$\rho^{-1} \ \approx \ \bar{\rho} \ \equiv \sum_{j=0}^{M} r_j b_j \tag{10}$$

then

$$\rho^{-1}\,\rho \ = \ 1 \quad \Rightarrow \quad \int_{\Omega} b_k \bar{\rho}\,\rho\, d\Omega \ = \ \int_{\Omega} b_k \, d\Omega \qquad k \ = \ 0,1,\,2,\ldots,\,M \tag{11}$$

This set of equations results in a system that is linear in r_j and is easily solved exactly. The flux terms, such as $(\rho u)(\rho v)/\rho$, can then be computed by multiplication, just as in the Taylor series approach. Alternatively, the projection method can be used to determine the flux expansion directly. Let

$$(\rho u)(\rho v)/\rho \ \approx \ F \ \equiv \sum_{j=0}^{M} b_j f_j \tag{12}$$

then

$$\int_{\Omega} b_k\,\rho F\, d\Omega \ = \ \int_{\Omega} b_k (\rho u)(\rho v)\, d\Omega \qquad k \ = \ 0,1,\,2,\ldots,\,M \tag{13}$$

defines a set of equations that are linear for f_j. This procedure is also applicable to flux functions that can be evaluated exactly by multiplication, such as in the case of the nonlinear Burger's equation. Let

$$\frac{U^2}{2} \ \approx \ F \ \equiv \sum_{l=0}^{M} b_l f_l \tag{14}$$

then

$$\int_{\Omega} b_k F\, d\Omega \ = \ \int_{\Omega} b_k \frac{U^2}{2}\, d\Omega \qquad k \ = \ 0,1,\,2,\ldots,\,M \tag{15}$$

The projection approach is appealing because the error of the approximate flux is uniformly minimized over the whole of the element, whereas in the Taylor series approach, the flux will be most accurate near the center of the element and less accurate near the edges of the element.

Volume Integral

After the flux is represented as in equation (9), the volume integral of equation (6) can easily be rewritten as a matrix times a vector as

$$J\mathbf{M}\,\mathbf{V}_t \ - \ \vec{\mathbf{G}}\cdot\vec{\mathbf{F}} + \int_{\partial\Omega} [b_k]\,\mathbf{J}^{-1}\vec{F}^R \ \cdot \ J d\vec{s} \ = \ 0 \tag{16}$$

where $\mathbf{V} \ = \ [v_k]$, $\vec{\mathbf{G}} \ = \ [\vec{g}_{k,j}]$, $\vec{\mathbf{F}} \ = \ \left[\vec{\bar{f}}_j\right]$,

$$\vec{g}_{k,j} \ = \ \int_{\mathbf{\Omega}} (\nabla b_k)\, b_j \, d\Omega, \qquad \vec{\bar{f}}_j \ = \ J\,\mathbf{J}^{-1}\,\vec{f}_j \qquad k \ = \ 0,1,2,\ldots,N \qquad j \ = \ 0,1,2,\ldots,M \tag{17}$$

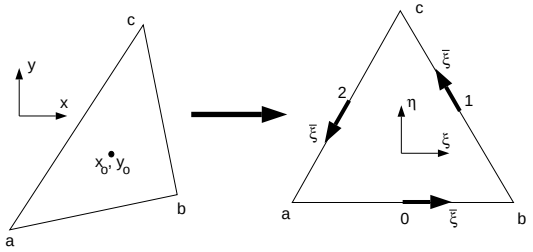
and $M \ \geq \ N$ is the degree to which a nonlinear flux is expanded in the basis set. The vector matrix $\vec{\mathbf{G}}$ is a constant for all elements of a given family and can be precomputed and stored. The evaluation of the volume integral becomes an order $M+1$ operation for each of the $N+1$ equations in the element.

Boundary Integral

The boundary integral is partitioned into segments associated with the sides of the element. The integral on each boundary segment can be rewritten in terms of a vector times a precomputed matrix; however, the procedure is complicated by the fact that $\vec{F}^R$ is a function of the solution in the two elements on either side of a boundary segment and each of these elements has a distinct local coordinate system. The remedy is to translate each basis function from their local coordinate systems to a coordinate system that is common to both elements. The use of a boundary-segment-based coordinate system actually results in a reduction in the total work and storage required. The complete process can be summarized in three steps: 1) translate the solution to an edge coordinate system, 2) compute the approximate Riemann flux in the edge coordinates, 3) project the flux onto the space defined by the element basis set.

For the sake of clarity, the procedure is illustrated for a triangular element; however, the same procedure can be applied to all types of elements, and the procedure facilitates the use of mix element types (i.e.,

squares and triangles together). Figure 1 illustrates a general triangle that has been mapped into a equilateral triangle. The equilateral triangle has a local coordinate system $(\xi,\ \eta)$ with its origin at the centroid of the element; each edge also has a local coordinate $\bar{\xi}$ with its origin in the center of the edge. Note that the dimen-



$$\begin{bmatrix} x - x_o \\ y - y_o \end{bmatrix} \;=\; \mathbf{J} \begin{bmatrix} \xi \\ \eta \end{bmatrix} \tag{18}$$

Figure 1. Transform from general triangle to equilateral similarity triangle that shows coordinate systems associated with interior and edges of similarity triangle.

sionality of the edge coordinate system is one lower than that of the element. Hence, for a three-dimensional element the edge coordinate system is two dimensional; for a two-dimensional element, the edge coordinate system is one dimensional; and for a one-dimensional element, the edge coordinate system contracts to a single point.

For each edge of the triangle, a constant matrix $\mathbf{T}_j$ exist that relates each member of the local basis to an expansion in terms of the edge coordinate. The subscript j identifies the edge to which the matrix applies. For example, on edge number 0,

$$\begin{aligned} \mathbf{B} \;\equiv [b_l] \;&=\; \begin{bmatrix} 1 \\ \xi \\ \eta \\ \xi^2 \\ \xi\eta \\ \eta^2 \\ \vdots \end{bmatrix} \\ &=\; \begin{bmatrix} 1 & 0 & 0 & \dots \\ 0 & 1 & 0 & \dots \\ \frac{-1}{2\sqrt{3}} & 0 & 0 & \dots \\ 0 & 0 & 1 & \dots \\ 0 & \frac{-1}{2\sqrt{3}} & 0 & \dots \\ \frac{1}{12} & 0 & 0 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} 1 \\ \bar{\xi} \\ \bar{\xi}^2 \\ \vdots \end{bmatrix} \\ &=\; \mathbf{T}_0 \left[\bar{b}_j\right] \;\equiv\; \mathbf{T}_0\, \bar{\mathbf{B}} \end{aligned} \tag{19}$$

The matrices for the other two edges are considerably more dense but are easily derived in exact form with the aid of a symbolic algebra package such as Maple or Mathematica. The $\mathbf{T}_j$ matrix for square elements is relatively sparse on all four edges.

Given any function expanded in terms of the element basis, the expansion in terms of the edge basis $\bar{\mathbf{B}}$ is derived as follows:

$$\begin{aligned} V_i \;&=\; \sum_{j=0}^{N} v_{i,j}\, b_j \;=\; \mathbf{V}_i^T\, B \\ &=\; \mathbf{V}_i^T\, \mathbf{T}_j\, \bar{\mathbf{B}} \;=\; \left[\mathbf{V}_i \mathbf{T}_j^T\right]^T \bar{\mathbf{B}} \;\equiv\; \bar{\mathbf{V}}_{i,j}^T\, \bar{B} \end{aligned} \tag{20}$$

hence,

$$\bar{\mathbf{V}}_{i,j} \;=\; \mathbf{T}_j^T\, \mathbf{V}_i \tag{21}$$

After $\bar{\mathbf{V}}_{i,j}$ has been computed on every face of every element, the flux through an edge can be computed without regard to the type of elements or the orientation of the coordinate system of the elements that border the edge. At an arbitrary edge, illustrated in figure 2, we arbitrarily designate one element to be on the left and the other to be on the right. The two edge coordinate

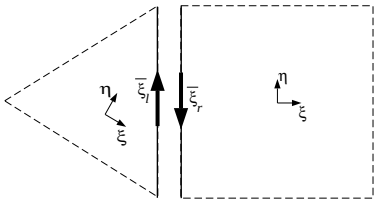


Figure 2. Relation of edge coordinates of adjacent elements.

systems of the two elements always point in opposite direction. A function of the edge coordinates of the right element can be represented in terms of the edge coordinate system of the left element simply by negating the odd members of $\bar{\mathbf{V}}_{i,j}$. In the three-dimensional case, the relation is not as trivial because a rotation may also be involved.

In the present work, the Riemann flux is approximated by a simple Lax–Friedrichs flux of the form

$$\begin{aligned} J\,\mathbf{J}^{-1}\vec{F}_l^R \cdot d\vec{s} \;\equiv\; \frac{1}{2}\Big\{ \Big[J\,\mathbf{J}^{-1}\Big(\vec{F}_l \;+\; \vec{F}_r\Big)\Big] \cdot \vec{n} \\ - \alpha[V_r \;-\; V_l]\Big\} ds \end{aligned} \tag{22}$$

where $d\vec{s} = \vec{n} ds$, the subscripts *l* and *r* denote the left and right sides of the edge, $\vec{n}$ points from left to right, and α is some smooth positive function that is greater in magnitude that the eigenvalues of the Jacobian of $\frac{1}{2}\Big[J\,\mathbf{J}^{-1}\Big(\vec{F}_l \;+\; \vec{F}_r\Big)\Big] \cdot \vec{n}$. By applying equation (21), the Lax–Friedrichs flux is easily expressed in terms of the edge coordinate to give

$$J\,\mathbf{J}^{-1}\vec{F}_l^R \cdot d\vec{s} \;=\; \bar{\mathbf{F}}_l^{\mathrm{T}}\, \bar{\mathbf{B}}\, ds \tag{23}$$

where

$$\begin{aligned}\bar{\mathbf{F}}_l \;&\equiv\; \left[\bar{f}_0, \bar{f}_{,1}\bar{f}_{,2}\bar{f}_{,3}\ldots\right] \\ &=\; \Big\{\Big[J\,\mathbf{J}^{-1}\Big(\mathbf{T}_{j_l}\,\vec{\mathbf{F}}_l \;+\; \widehat{\mathbf{I}}\,\mathbf{T}_{j_r}\,\vec{\mathbf{F}}_r\Big)\Big]\cdot\vec{n} \\ &\quad -\alpha\Big[\widehat{\mathbf{I}}\,\mathbf{T}_{j_r}\,\mathbf{V}_r \;-\; \mathbf{T}_{j_l}\,\mathbf{V}_l\Big]\Big\}/2 \\ &=\; \Big\{\Big[J\,\mathbf{J}^{-1}\Big(\vec{\bar{\mathbf{F}}}_l \;+\; \widehat{\mathbf{I}}\,\vec{\bar{\mathbf{F}}}_r\Big)\Big]\cdot\vec{n} \\ &\quad -\alpha\Big[\widehat{\mathbf{I}}\,\bar{\mathbf{V}}_r \;-\; \bar{\mathbf{V}}_l\Big]\Big\}/2 \end{aligned} \tag{24}$$

and $\widehat{\mathbf{I}} \;=\; diag(1,\,-1,\,1,\,-1,\,\ldots)$ accounts for the difference in the left and right edge coordinates. The coefficients of the approximate Riemann flux on the right face are simply $\bar{\mathbf{F}}_r \;=\; \widehat{\mathbf{I}}^{-1}\,\bar{\mathbf{F}}_l$. As a side note, depending on the form of the flux, less computational effort by be required to compute $\vec{\bar{\mathbf{F}}}$ directly from $\bar{\mathbf{V}}$ than to translate the flux from element to edge coordinates. This is especially true for linear fluxes.

Finally, the boundary integral is evaluated in the edge frame of reference by expressing b_k in terms of the edge coordinates and collecting in terms of the components of $\bar{\mathbf{F}} \;=\; \left[\bar{f}_0, \bar{f}_1, \bar{f}_2, \ldots\right]$. To illustrate, let $[t_k]_j$ denote the kth row of $\mathbf{T}_j$. The integral on a boundary segment becomes

$$\begin{aligned}\int\limits_{\partial\Omega} (b_k)\,\left(\bar{\mathbf{F}}^T\,\bar{\mathbf{B}}\right)\,ds \;&=\; \int\limits_{\partial\Omega}\Big([t_k]_j\,\bar{\mathbf{B}}\Big)\,\left(\bar{\mathbf{F}}^T\,\bar{\mathbf{B}}\right)\,ds \\ &=\; \int\limits_{\partial\Omega}\Big\{\left(t_{k,0}+t_{k,1}\bar{\xi}+t_{k,2}\bar{\xi}^2+t_{k,3}\bar{\xi}^3+\ldots\right) \\ &\qquad\cdot\left(\bar{f}_0+\bar{f}_1\bar{\xi}+\bar{f}_2\bar{\xi}^2+\bar{f}_3\bar{\xi}^3+\ldots\right)\Big\}\,ds \\ &=\; \int\limits_{\partial\Omega}\Big\{\left(t_{k,0}+t_{k,1}\bar{\xi}+t_{k,2}\bar{\xi}^2+\ldots\right)\bar{f}_0 \\ &\qquad+\;\left(t_{k,0}\,\bar{\xi}+t_{k,1}\bar{\xi}^2+t_{k,2}\bar{\xi}^3+\ldots\right)\bar{f}_1 \\ &\qquad+\;\left(t_{k,0}\,\bar{\xi}^2+t_{k,1}\bar{\xi}^3+t_{k,2}\bar{\xi}^4+\ldots\right)\bar{f}_2 \\ &\qquad+\;\ldots\Big\}\,ds \\ &\equiv\; [e_k]\,\bar{\mathbf{F}}\end{aligned} \tag{25}$$

where $[e_k]$ is a constant row matrix that is easily evaluated exactly. Let $\mathbf{E}$ denote the matrix that is generated by applying the above process to each member of the basis set. The final form of the semidiscrete equation is

$$\mathbf{V}_t \;-\; \mathbf{J}^{-1}\left[\mathbf{M}^{-1}\vec{\mathbf{G}}\cdot\,\vec{\mathbf{F}} \;-\sum_{k=1}^{n_e}\left(\mathbf{M}^{-1}\mathbf{E}_k\,\bar{\mathbf{F}}_k\right)\right] \;=\; 0 \tag{26}$$

where n_e is the number of edges and the matrices $\mathbf{M}^{-1}\,\vec{\mathbf{G}}$ and $\mathbf{M}^{-1}\,\mathbf{E}_k$ are constant matrices that apply to all elements of a given type. Furthermore, these matrices can be efficiently precomputed by the procedure just described.

Computational Effort

The effort required to evaluate the complete spatial operator is contained in three basic operations: the evaluation of equation (26) for each element, the evaluation of equation (21) for each edge of each element, and the computation of $\vec{\mathbf{F}}$ and $\bar{\mathbf{F}}$ from $\mathbf{V}$ and $\bar{\mathbf{V}}$, respectively. The operation count of the first two operations is directly related to the size of the matrices $\mathbf{M}^{-1}\,\vec{\mathbf{G}}$, $\mathbf{M}^{-1}\,\mathbf{E}_k$, and $\mathbf{T}_j$. The row dimension of all three matrices is $N(p,d)+1$. If the flux is linear, then the column space of $\mathbf{M}^{-1}\,\vec{\mathbf{G}}$ is also $N(p,d)+1$. However, in the nonlinear case the flux must be expanded to at least degree $p+1$; and thus, the column dimension must be at least $N(p+1,d)+1$. The column dimension of both $\mathbf{M}^{-1}\,\mathbf{E}_k$ and $\mathbf{T}_j$ is $N(p,d-1)+1$.

The operation count of the flux computation can vary considerably depending on the complexity of the flux function. In the linear case the operation count is on the order of $N(p,d)+1$ and $N(p,d-1)+1$ for $\vec{\mathbf{F}}$ and $\bar{\mathbf{F}}$, respectively. The operation count in the nonlinear case could be as high as $\{N(p,d)+1\}\,\{N(p+1,d)+1\}$ and $\{N(p,d-1)+1\}^2$ for $\vec{\mathbf{F}}$ and $\bar{\mathbf{F}}$, respectively. Thus, the total operation count for the spatial operator in a single element varies from

$$\begin{aligned}&d[N(p,d)+1]^2 \\ &+\;2n_e[N(p,d)+1][N(p,d-1)+1] \\ &+\;O(d[N(p,d)+1]\;+\;n_e[N(p,d-1)+1])\end{aligned} \tag{27}$$

for the linear case to

$$\begin{aligned}&d[N(p,d)+1][N(p+1,d)+1] \\ &+\;2n_e[N(p,d)+1][N(p,d-1)+1] \\ &+\;O\{d[N(p+1,d)+1][N(p,d-1)+1] \\ &\qquad+\;n_e[N(p,d-1)+1]^2\}\end{aligned} \tag{28}$$

for the nonlinear case. These estimates assume that the matrices are full, which is not the case. The exact form of $N(p,d)$ is

$$N(p,d) \;=\; dp \;+\; \begin{cases}0 & d=1\\ p(p-1)/2 & d=2\\ 3p(p-1)/2 & d=3\\ 3p(p-1) & d=4\end{cases} \tag{29}$$

and Table 1 gives $N(p,d)+1$ for a degree range of $1 \le p \le 6$ and for time-space dimensions that range from 1 to 4.

In a conventional DG implementation (i.e. one that uses quadrature points), the operation count is on the order of

$$\begin{aligned}&[N(p,d)+1]\{(1+d)N_{qv}\;+\;2n_eN_{qb}\} \\ &\quad+\;O(dN_{qv}\;+\;n_eN_{qb})\end{aligned} \tag{30}$$

where N_{qv} and N_{qb} denote the number of quadrature points required for the volume integral and boundary segment integrals, respectively, and the last term denotes the cost of computing the flux at each quadrature point. Most references do not give the specific quadrature formulas used; however, Halt et al[6] referred to the work of Dunavant[12] who derived nearly optimal formulas in which $N_{qv} > N(p,d)+1$ in order to evaluate the integral exactly to degree $2p$. Thus, the operation count for the conventional DG implementation is greater than the values given by either (27) or (28) even if the sparseness of the matrices is not taken advantage of.

When the DG method is compared to fundamentally different methods such as finite-difference or finite-volume methods, the comparison must be done in an equitable manner. To do so, we hypothesize that any two methods that have the same degree of accuracy and the same physical stencil size will give similar results (for benign cases that do not violate the basic assumptions of the method). In practice, we compare methods that are of the same order of accuracy and have the same total number of variables. In this frame of reference, the evaluation of the spatial operator is an operation of order $N(p,d)+1$ or $N(p+1,d)+1$ per dependent variable for a linear or nonlinear problem, respectively.

Time Integration and Stability

The solution is advanced in time with a three-stage TVD Runge-Kutta method:[13]

$$\begin{aligned} W^0 &= V^{n-1} \\ W^k &= \beta_k W^0 + (1-\beta_k)\left[W^{k-1} + \Delta t R\left(W^{k-1}\right)\right] \\ & \qquad\qquad k = 1,2,3 \\ V^n &= W^3 \end{aligned} \tag{31}$$

where β_k = 0, 3/4, and 1/3 for k = 1, 2, and 3, respectively.

Fourier stability analysis has been applied to (31) for the one-dimensional linear case of

$$U_t + a\,U_x = 0 \tag{32}$$

to determine the stability limit $\lambda_k \equiv a\,\Delta t/\Delta x$ for methods of various orders (i.e., various values of p). The results given in Table 2 are for K-stage/Kth-order Runge-Kutta methods of the type described above, where K = 1, 2, and 3 and p ranges from 0 to 11. The rapid drop in the stability limit as the order of the method is increased would normally be alarming in comparison with stability constraints of explicit finite-difference methods. However, if we again require that comparison be made among methods having the same total number of variables, then the size of the element in the DG method would be larger (by a factor of $p+1$ in one-dimension) than the mesh size of a comparable finite-difference calculation. Thus, most of the drop in the stability limit can be attributed to definition of Δx. The right-most column of Table 2 gives $\lambda_k(p+1)$, which gives the DG stability limit in a form that facilitates comparison with the stability limit of a finite-difference method.

Results

One-Dimensional Test

The one-dimensional version of this method has been tested on the linear problem

$$U_t + a\,U_x = 0 \tag{33}$$

and the nonlinear problem

$$U_t + \frac{1}{2}\left(U^2\right)_x = 0 \tag{34}$$

on the domain $0 < x < 1$ with periodic boundary conditions.

A linear problem is solved first with smooth initial conditions $U(0,x) = \frac{1}{2} + \sin(2\pi x)$ to demonstrate the general accuracy properties of the method. The numerical solution is initialized by expanding the initial condition in a Taylor series about the center of each element. All components of the numerical solution are compared with the Taylor series of the exact solution after it has advected for several periods. The L_n-norm of the error of the *j*th component of the solution is defined as

$$L_{\mathrm{n}}(\varepsilon_j) \equiv \left[\left(\sum_{i=1}^{I} |v_{i,j} - u_{i,j}|^n\right)/I\right]^{1/n}$$

where $u_{i,j}$ denotes the Taylor coefficient of the exact solution in cell *i* and *I* denotes the number of elements. A mesh-refinement study has been performed for p = 1 through 5. The time step was chosen to be sufficiently small such that the error would be dominated by the spatial operator; however, for $p > 2$ the time step varied as $\Delta t \propto (\Delta x)^{(p+1)/3}$ so that the temporal accuracy would be of the same order as the spatial accuracy. Figure 3 shows the L_1-norm of the error for each component of the solution for p = 1, 2, and 4. The convergence rate of the solution between the two finest grids is given in Table 3. Although most cases converge at the design rate of $p+1$, the v_0 term of the $p = 1$ case converges at a rate of ≈ 3, which is one order higher than expected. This faster convergence is fortuitous and occurs only because the basis functions are incidentally orthogonal. Because v_1

is only second order and v_2 is undefined, a solution of degree > 1 cannot be recovered at any point other than the element center without departing from the Galerkin framework. Furthermore, although v_0 converges faster than the design order, its error is still considerably larger than the error of the p=2 case.

In the second test case, the DG method is applied to the linear problem with a discontinuous initial solution:

$$U(0,x) \;=\; \begin{cases} 1 & \frac{1}{4} \;<\; x \;<\; \frac{3}{4} \\ 0 & x \;\leq\; \frac{1}{4},\;\; x \;\geq\; \frac{3}{4} \end{cases} \qquad (36)$$

Figures 4 through 8 show several results for this case on a grid with 40 elements. Figures 4, 5, and 6 show the solution after one period for p = 1, 2, and 6, respectively. Each method has small overshoots that are confined to the neighborhood of the discontinuity. Similar results were observed for orders up to p = 11. Figures 7 and 8 also show solutions for p = 6, but the solution has advected for 5 and 50 periods, respectively. The overshoots neither grow or spread in time, which is in sharp contrast to the behavior of more traditional methods. A typical finite-difference approach, for instance, would tend to smear a contact discontinuity over a region that grows linearly in time.

Next, the DG method is applied to a linear test case that was prescribed as part of the ICASE/LaRC Workshop on Benchmark Problems in Computational Aeroacoustics[14]; these results are compared with the finite-difference results described in reference 14. The test case consists of a Gaussian pulse that is advected across a uniform domain. The Gaussian pulse has a half-width of 6 and is initially centered on the origin of a domain that ranges from –20 to 450. Results are shown in Figure 9 for p = 1, 2, and 3; however, as p is increased the number of elements is decreased such that the total number of variables is approximately 470 (the number of points specified in the workshop). In figure 10, the results of the fourth-order DG method at t = 400 are compared in detail with the results of a fourth- and fifth-order finite-difference method. The fourth-order DG method with only 117 elements is considerably better than either fourth- and fifth-order finite-difference methods using 470 points. (Note: smooth curves are generated for results of the DG method by evaluating the solution at several points within each element.)

The last one-dimensional test case is a nonlinear problem (eq. (34)) in which a shock forms from an initially smooth solution. This problem was used to not only demonstrate the robustness of the method but also to investigate the effect of truncating the nonlinear flux at various levels. We expect, based on the formulation, that the nonlinear flux must be expanded to $M = N(p+1,d)$ terms such that the degree of $\nabla b_k \cdot \vec{F}$ is the same as the degree of $b_k\, V$ to obtain the design rate of convergence of $p+1$. This expected convergence property was verified by a mesh refinement study in which the calculation was stopped just before shock formation. The mesh-refinement was performed for several values of p, and the finest grid contained 320 elements. Typical results, shown in Table 4 for $p = 2$, indicate that the convergence rate measured in the L_∞ norm drops to $\approx p$ when $M = N(p,d)$ but is $\approx p+1$ for all other cases.

Figure 11 shows solutions for $p = 1$ and $p = 2$ (second and third order) in which the shock has formed and has begun to propagate. In both cases, the solutions were obtained without the use of limiters, added dissipation, or entropy correction terms; however, the case in which $p = 2$ required that the nonlinear flux be fully expanded ($M = N(2p,d)$). Otherwise, the solution would diverge shortly after shock formation. All higher order cases ($p > 2$) required some type of limiter; work is continuing in this area.

Two-Dimensional Test

The DG method is applied to the scalar advection equation in two dimensions to demonstrate its robust treatment of unstructured grids. The test problem is given by

$$\begin{aligned} U_t \;+\; a\,U_x \;+\; b\,U_y \;&=\; 0 \\ U(0,x,y) \;&=\; [\sin(\pi x)\sin(\pi y)]^4 \end{aligned} \qquad (37)$$

defined on the periodic domain $0 < x,y < 1$. The approximate solution V_i is initialized from the Taylor expansion of the exact initial condition. The baseline case is chosen to be a uniform Cartesian grid that is triangulated in a regular manner, as illustrated in Figure 12. Figure 13 shows the L_1 norm of the error in the v_0 component of the solution. As in the one-dimensional case, the time step was small so that the spatial error dominated and, for $p > 2$, the time step was proportional to $(\Delta x)^{(p+1)/3}$. In mesh refinement studies, the first grid in the sequence is coarsened as the order of the method is increased so that the total number of variables is roughly the same. The abscissa in figure 13 is the square root of the total number of variables, which facilities comparison with a simple fourth-order finite-difference method. The higher-order convergence of the $p = 1$ case that was observed in one dimension is not observed in the two-dimensional case. The accuracies of the fourth-order DG and finite-difference methods are quite similar.

One of the major motivations for pursuing a DG method is its ability to maintain accuracy for complex geometries. Here, the baseline grid is altered in several

ways to test and demonstrate this capability. Figure 14 illustrates four of the variations that were tested. In the first case, grid A is uniform like the baseline case but the triangulation has been done in a random manner. In the second case, grid B is generated from grid A by smoothly clustering the grid toward a diagonal. Grid C is generated from grid B by randomly perturbing each grid point by an amount that is less than 20 percent of the average mesh size. In the last case, grid D is generated from the baseline by imposing a piecewise-constant mesh spacing that places half of the points in a narrow band around the axis.

In most cases, the measured error was found insensitive to the grid modification or the direction of propagation (i.e., the value of a and b). Figure 15 gives the L_1 norm of the error for $p = 3$ (fourth-order) on each of the grids shown in figure 14. Results for other values of p were similar. All results in Figure 15 are for $a = 1$, $b = 0$, except for case D2 where $a = 1$ and $b = 1$. The slight increase in error in case D2 is attributed more to the increase in mesh size along the propagation path than to the discontinuous manner in which it changes.

In the last test case, the DG method is applied to another problem prescribed as part of the ICASE/LaRC Workshop on Benchmark Problems in Computational Aeroacoustics.[14] The linear Euler equations are solved on a square domain of dimensions $-100 < x, y < 100$ with initial conditions that place a compact acoustic source at $x = y = 0$ and a convecting disturbance at $x = 67$, $y = 0$. Here, the equations have been recast in a form that emphasizes the decoupling of the convection terms from the acoustic terms that occurs in this linear system.

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{E}}{\partial x} + \frac{\partial \mathbf{F}}{\partial y} = 0 \qquad (38)$$

where

$$\mathbf{U} = \begin{bmatrix} \rho - P \\ p \\ u \\ v \end{bmatrix}$$

$$\mathbf{F} = \begin{bmatrix} M_x(\rho - P) \\ M_x P + u \\ M_x u + P \\ M_x v \end{bmatrix}, \qquad \mathbf{G} = \begin{bmatrix} M_y(\rho - P) \\ M_y P + v \\ M_y u \\ M_y v + P \end{bmatrix}$$

$M_x = 0.5$, $M_y = 0$, and

$$(\rho - P)(0, x, y) = 0.1 \exp\left[(-\ln(2))\frac{(x-67)^2 + y^2}{25}\right]$$

$$P(0, x, y) = \exp\left[(-\ln(2))\left(\frac{x^2 + y^2}{9}\right)\right]$$

$$u(0, x, y) = 0.04\, x\, P(0, x, y)$$

$$v(0, x, y) = 0.04\, y\, P(0, x, y)$$

The workshop, which targeted finite-difference methods, prescribed a grid of $200\,X\,200$. In the present calculation, we use a uniform $n\,X\,n$ Cartesian grid that has been randomly triangulated (as shown in Figure 14a in which the number of elements is chosen as a function of the degree p such that the total number of unknowns equals approximately 200^2. Results are shown for p = 0, 1, 2, and 3 (first, second, third, and fourth order) with n = 141, 81, 57, and 44, respectively. Figure 16 shows P and u at $t = 40$ for the $p = 3$ case. The wave fronts appear smooth and cylindrical in spite of the fact that the initial disturbance was smaller than the element size. A more quantitative comparison is shown in Figures 17 and 18. The pressure P is plotted on the $x = 0$, $t = 40$ line for p = 0 through 3 at the resolution given above. Also shown is a fine-grid solution with p = 3, n = 132 and the solution from a fifth-order finite-difference method on a $200\,X\,200$ grid. An enlargement of the right peak (Figure 18) shows that all solutions that are third order or better give similar results.

Summary

A quadrature-free form of the discontinuous Galerkin method has been formulated for the hyperbolic conservation laws. This approach reduces both the storage and operation count to levels that are comparable to high-order finite-volume methods. The method is well suited to both unstructured and structured grids and it has been tested on several one- and two-dimensional problems to demonstrate its accuracy and robustness. On smooth meshes, the accuracy of the DG method is comparable to or better than traditional high-order finite-difference methods. Contact discontinuities are advected without the usual diffusion effect, and nonlinear discontinuities (shocks) are propagated by second- and third-order methods without the use of limiters. On two-dimensional unstructured grids, random and discontinuous mesh variations had little effect on the error and no effect on the convergence of the error.

References

1. C. Johnson and J. Pitkärata, "An Analysis of the Discontinuous Galerkin Method for a Scalar Hy-

perbolic Equation," Mathematics of Computation, v46 (1986), pp. 1–26.
2. B. Cockburn and C.-W. Shu, "TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: general framework," Mathematics of Computation, v52 (1989), pp. 411-435.
3. B. Cockburn, S.Y. Lin and C.-W. Shu, "TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: one dimensional systems," Journal of Computational Physics, v84 (1989), pp. 90-113.
4. B. Cockburn, S. Hou and C.-W. Shu, " The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: the multidimensional case," Mathematics of Computation, v54 (1990), pp. 545-581.
5. G. Jiang and C.-W. Shu, " On cell entropy inequality for discontinuous Galerkin methods," Mathematics of Computation, v62 (1994), pp. 531-538.
6. D. W. Halt and R. K. Agarwal, "Compact Higher Order Characteristic-Based Euler Solver for Unstructured Grids," AIAA J. v30 (1992), pp. 1993–1999.
7. F. Bassi and S. Rebay, "Accurate 2D Euler Computations by means of a High-Order Discontinuous Finite Element Method," Proceedings of the 14th International Conference on Numerical Methods in Fluid Dynamics, Bangalor, India, July 11–15, 1994.
8. F. Bassi and S. Rebay, "Discontinuous Finite Element High Order Accurate Numerical Solution of the Compressible Navier-Stokes Equations," Presented at the ICFD Conference on Numerical Methods in Fluid Dynamics, University of Oxford, Oxford, England, April 3–6, 1995
9. R. B. Lowrie, P. L. Roe, and B. van Leer, "A Space-Time Discontinuous Galerkin Method for the Time-Accurate Numerical Solution of Hyperbolic Conservation Laws," presented at the 12th AIAA Computational Fluid Dynamics Conference, San Diego, CA, June 19–22, 1995
10. R. Biswas, K.D. Devine, and J. Flaherty, "Parallel, adaptive finite element methods for conservation laws," Applied Numerical Mathematics, v14 (1994), pp. 255–283.
11. C.A.J. Fletcher, *Computational Galerkin Methods* Springer-Verlag, New York, 1984.
12. D.A. Dunavant, "High Degree Efficient Symmetrical Gaussian Quadrature Rules for the Triangle, " International Journal for Numerical Methods in Engineering, Vol. 21, pp. 119–1148, 1985.
13. C.-W. Shu and S. Osher, "Efficient implementation of essentially non-oscillatory shock-capturing schemes," Journal of Computational Physics, v77 (1988), pp. 361-383.
14. Harold L. Atkins, "Application of Essentially Nonoscillatory Methods to Aeroacoustic Flow Problems," *Proceedings of ICASE/LaRC Workshop on Benchmark Problems in Computational Aeroacoustics*, Edited by J.C. Hardin, J.R. Ristorcelli, and C.K.W. Tam, NASA Conference Publication 3300, May, 1995, pp. 15-26

Table 1 $N(p,d)+1$ for specific values of p and d

p	d			
	1	2	3	4
1	2	3	4	5
2	3	6	10	15
3	4	10	19	31
4	5	15	31	53
5	6	21	46	81
6	7	28	64	115

Table 2 $\lambda_k \equiv a\,\Delta t\,/\,\Delta x$ for
K-stage/*K*th-order Runge-Kutta methods
applied to a DG method of order p

p	λ_1	λ_2	λ_3	$\lambda_3 * (p+1)$
0	1.0	1.00	1.256	1.256
1	0.001	0.333	0.409	0.818
2	u-s‡	0.06	0.209	0.627
3	u-s	0.02	0.13	0.52
4	u-s	0.01	0.089	0.445
5	u-s	0.006	0.066	0.396
6	u-s	0.004	0.051	0.306
7	u-s	0.003	0.04	0.32
8	u-s	0.002	0.033	0.297
9	u-s	0.002	0.027	0.27
10	u-s	0.001	0.023	0.253
11	u-s	0.001	0.02	0.24

‡ u-s denotes unstable method

Table 3 Convergence Rates of $L_1(\varepsilon_j)$ between two finest grids

p	j				
	0	1	2	3	4
1	2.990	2.133			
2	3.064	3.049	3.022		
4	4.99	5.74	5.00	5.00	5.00

Table 4 Effect of truncating the nonlinear flux on grid convergence: p=2 case.

Norm of measure	M		
	$N(p,d)$	$N(p+1,d)$	$N(p+2,d)$
$L_1(\varepsilon_0)$	2.894	2.944	2.942
$L_\infty(\varepsilon_0)$	2.015	2.923	2.921

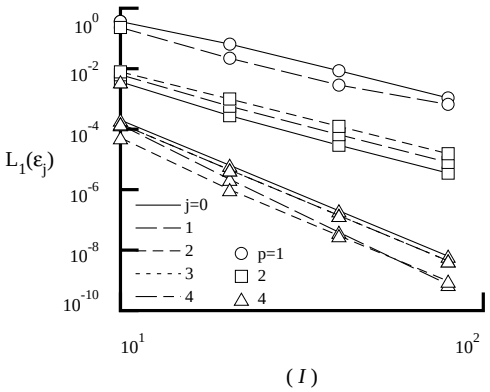


Figure 3. Convergence of the L_1 norm of the error for:

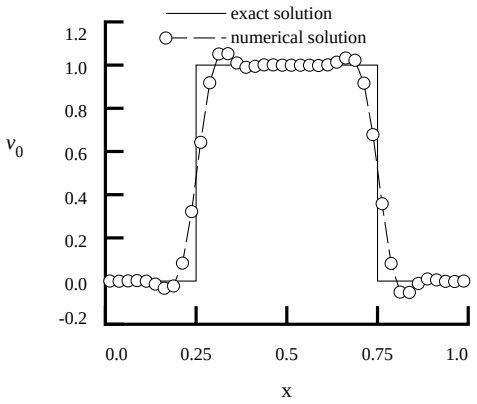


Figure 4. Solution of the linear problem after one time period given by DG method with p=1.

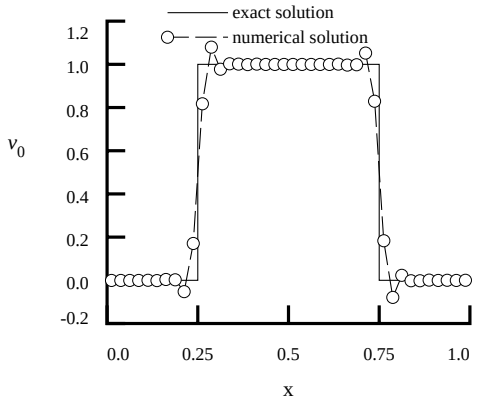


Figure 5. Solution of the linear problem after one time period given by DG method with p=2.

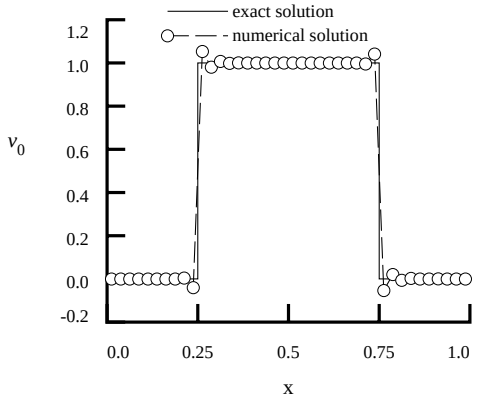


Figure 6. Solution of the linear problem after one time period given by DG method with p=6.

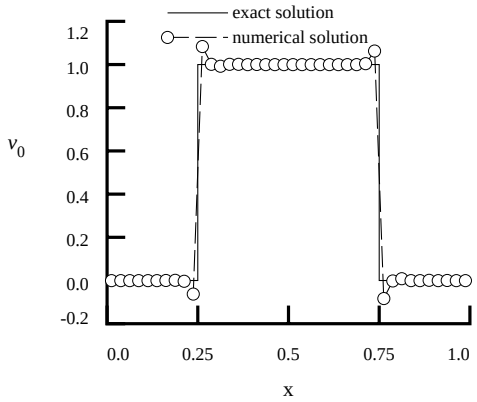


Figure 7. Solution of the linear problem after five time periods given by DG method with p=6.

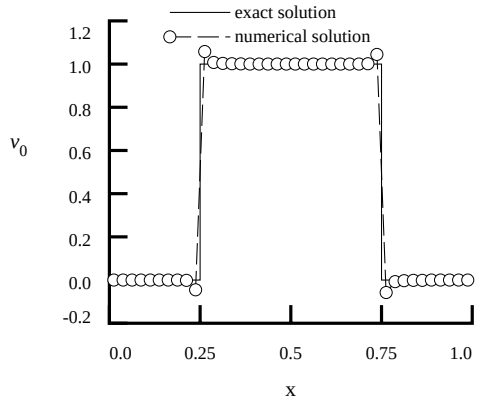


Figure 8. Solution of the linear problem after 50 time periods given by DG method with p=6

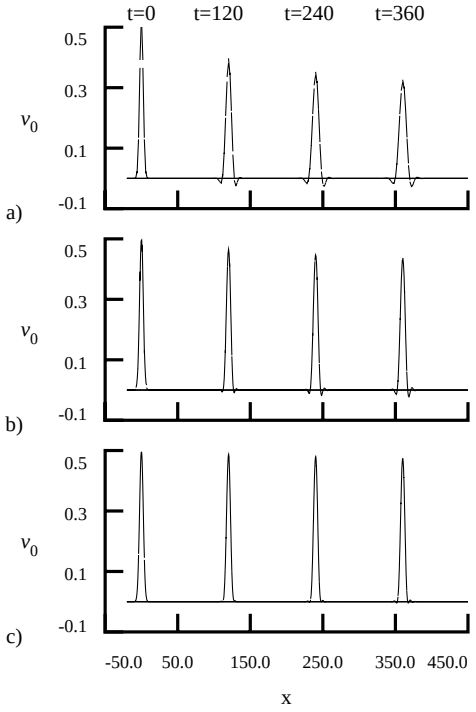


Figure 9. The DG method applied to the advection of a Gaussian pulse: a) p = 1 with 235 elements; b) p = 2 with 156 elements; and c) p = 3 with 117 elements

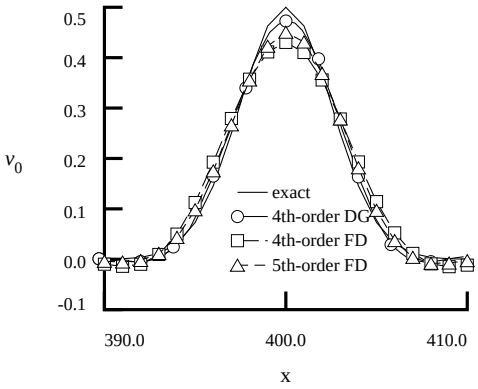


Figure 10. Comparison of fourth-order DG with fourth- and fifth-order finite-difference (FD)

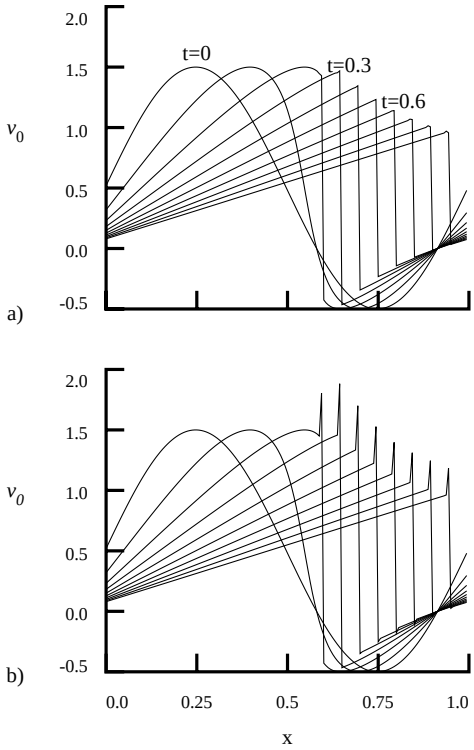


Figure 11. Solution of the nonlinear equation with a) p=1, b) p=2

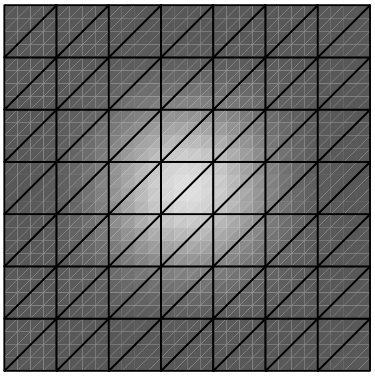


Figure 12. Triangulated grid and solution of scalar advection problem

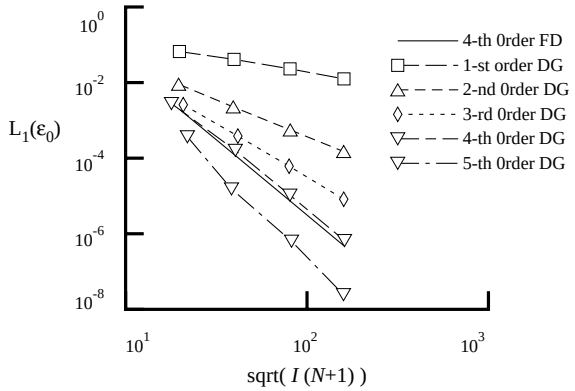


Figure 13. Convergence of solution for scalar advection on an unstructured grid for various orders of accuracy.

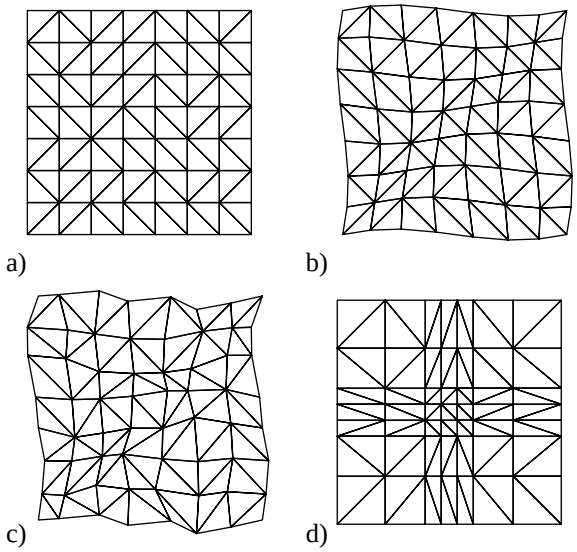


Figure 14. Variation on the baseline unstructured grid: a) random triangulation, b) smoothly clustered toward diagonal, c) random perturbation of 20 percent of the mean cell spacing, d) discontinuous mesh variation.

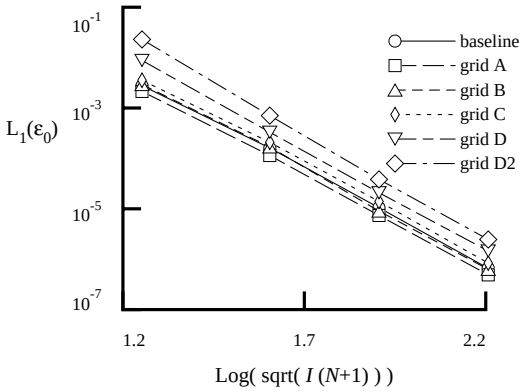


Figure 15. Convergence of solution for scalar advection for a fourth order method on several versions of the unstructured grid.

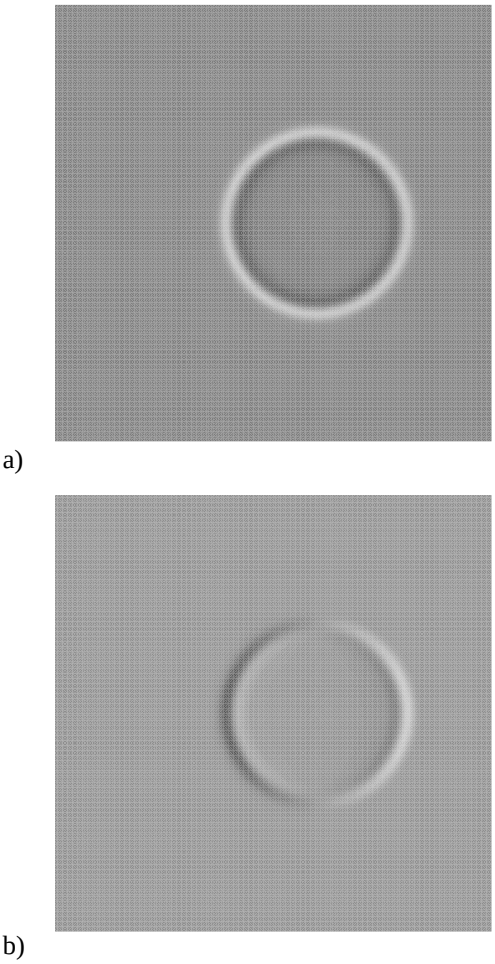


Figure 16. Acoustic wave modeled by the linear Euler Equations. $p = 3$, $t = 40$ for a) P, b) u.

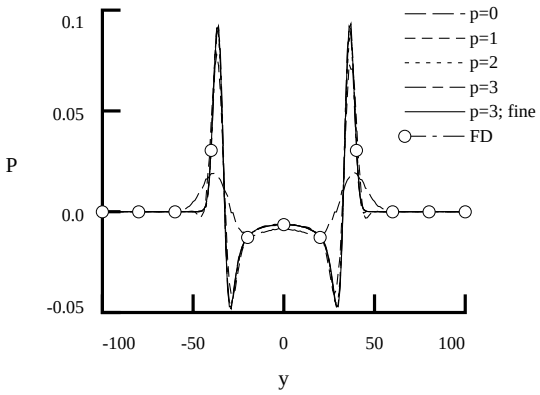


Figure 17. Solution of the linear Euler equations: Pressure on x=0 at t=40

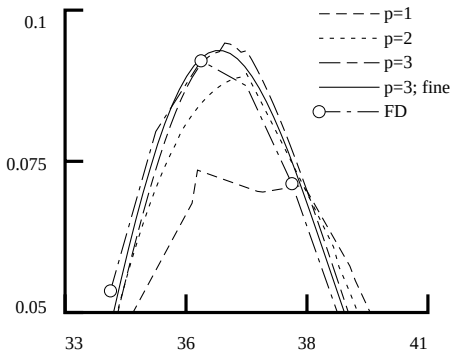


Figure 18. Solution of the linear Euler equations: Pressure on x=0 at t=40: enlargement of solution near y=36