The Spectral Difference Method for the 2D Euler Equations on Unstructured Grids

Z.J. Wang[*]
Department of Aerospace Engineering, Ames, IA 50011

and

Yen Liu[†]
NASA Ames Research Center, Moffett Field, CA 94035

An efficient, high-order, conservative method named the spectral difference method has been developed recently for conservation laws on unstructured grids. It combines the best features of structured and unstructured grid methods to achieve high computational efficiency and geometric flexibility; it utilizes the concept of discontinuous and high-order local representations to achieve conservation and high accuracy; and it is based on the finite-difference formulation for simplicity. The method is easy to implement since it does not involve surface or volume integrals. Universal reconstructions are obtained by distributing solution and flux points in a geometrically similar manner for simplex cells. In this paper, the method is further extended to nonlinear systems of conservation laws, the Euler equations. Accuracy studies are performed to numerically verify the order of accuracy. In order to capture both smooth feature and discontinuities, a monotonicity limiter is implemented, and tested for a double Mach reflection problem. The method is more efficient than the discontinuous Galerkin and spectral volume methods for unstructured grids.

I. Introduction

A new, high-order, conservative, and efficient method named the spectral difference (SD) method has been recently developed by Liu, Vinokur and Wang[1,2] for conservation laws on unstructured grids. In the present study, the SD method is further extended to the Euler equations. The primary motivation for developing another numerical method is to seek a simpler to implement and more efficient method than the current state of the art - the discontinuous Galerkin (DG) method[3-5], and the spectral volume (SV) method[6-10], to name just a few high-order methods for conservation laws on unstructured grids. As a matter of fact, the DG, SV and SD methods are similar in that they share the same the solution space, i.e., the space of piece-wise discontinuous polynomials, and some Riemann solvers[11,12] are used at the element interfaces to provide solution coupling between the discontinuous elements and appropriate numerical dissipation necessary to achieve stability. In addition, all of them are conservative locally at the element level, making them suitable for problems with discontinuities. They do differ on how solution unknowns or degrees-of-freedom (DOFs) are chosen, and how the DOFs are updated. In a DG method, the DOFs are either the expansion coefficients for a given set of polynomial basis functions or solutions at selected locations within the element. In a SV method, however, the DOFs are subcell mean solutions, while in the SD method, the DOFs are the solutions at (usually) the Gauss quadrature points. The difference between the DG, SV and SD methods is the same as the difference between the Galerkin finite element (FE), finite volume (FV), and finite difference (FD) methods.

In the DG method, a Galerkin finite-element method is employed to update the unknowns within each cell. This requires (usually) the inversion of a mass matrix, and the use of quadratures of roughly twice the order of accuracy of the reconstruction to evaluate the surface integrals for non-linear flux functions and additional volume integrals. In the SV method, the integral conservation law is used to update volume averages over subcells defined by a geometrically similar partition of each grid cell. As the order of accuracy increases, the partitioning for a 3D simplex cell requires the introduction of a large number of parameters, whose optimization to achieve convergence becomes

[*] Associate Professor of Aerospace Engineering, 2271 Howe Hall, zjw@iastate.edu, Associate Fellow of AIAA
[†] Research Scientist, Yen.Liu@nasa.gov, Mail Stop T27B-1, AIAA Member.

increasingly more difficult. Also, the large number of interior facets, and the additional increase in the number of quadrature points for each facet, increases the computational cost greatly. Because there are no volume or surface integrals in the SD method, it is easier to implement in multiple dimensions than the DG and SV methods. For the same reason, it will be shown later that the SD method is indeed much more efficient than the DG and SV methods.

In the spectral difference (SD) method, the number of DOFs in each cell is the number of nodal values required to support a reconstruction of a given order of accuracy. Their locations are chosen so that a quadrature approximation for the volume integral exists at least to the same order of accuracy. The fluxes are calculated at a different set of nodes, whose number will support a reconstruction of one order higher, since the flux derivatives are used to update the conservative unknowns. They are located so that quadrature approximations for surface integrals over the cell boundaries exist to a required order of accuracy. In addition, the locations of the solution points and the flux points must be such that the integral conservation law is satisfied for the cell to the desired order of accuracy. If the points are distributed in a geometrically similar manner for all cells, the reconstruction and discretizations become universal, and can be expressed as the same weighted sums of the products of the local metrics and fluxes. These metrics are constants for the line, triangle, and tetrahedron elements, and can be computed analytically for curved elements. We can also show that the number of flux points is less than the number of quadrature points in the SV method. Since all unknowns are decoupled, no mass matrix inversion is required.

Conventional unstructured finite-difference[13] and finite-volume (FV)[14-15] methods require data reconstructions based on the least-squares formulation using neighboring point or cell data. Since each unknown employs a different stencil, one must repeat the least-squares inversion for every point or cell at each time step, or store the inversion coefficients. In a high-order, three-dimensional computation, the former would involve impractically large CPU time, while for the latter the memory requirement becomes prohibitive. In addition, the finite-difference method does not usually satisfy the integral conservation in general. In contrast, the DG, SV and SD methods employ a local, universal reconstruction of a given order of accuracy in each cell in terms of internally defined conservative unknowns. This is the main reason why high-order DG, SV and SD methods are more efficient than a high-order finite volume method.

The SD formulation is similar to the pseudo-spectral or collocation spectral method[16] in that both employ nodal solutions as the DOFs and both formulations are based on the differential form of the governing equations. In fact, the multi-domain spectral method developed by Kopriva[17-18] and the SD method degenerate to a similar method in one dimension. The SD method can be viewed as an extension of the multidomain spectral method to a simplex unstructured grid.

The paper is organized as follows. In the next section, the basic idea of the SD method is presented in the physical domain. Its efficient implementation and conservation property are discussed in Section 3. Sample numerical results including a numerical accuracy study are presented in Section 4. Conclusions and possible future work are outlined in Section 5.

II. Basic Idea of the Spectral Difference Method

Consider the unsteady 2D Euler equations in conservative form written as

$$\frac{\partial Q}{\partial t}+\frac{\partial f}{\partial x}+\frac{\partial g}{\partial y}=0\,, \tag{1a}$$

where Q is the vector of conserved variables, f and g are the inviscid fluxes given below:

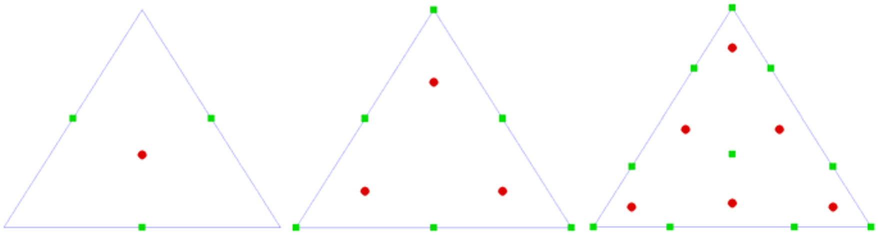
$$Q=\begin{Bmatrix}\rho\\ \rho u\\ \rho v\\ E\end{Bmatrix},\qquad f=\begin{Bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(E+p)\end{Bmatrix},\qquad g=\begin{Bmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ v(E+p)\end{Bmatrix}, \tag{1b}$$

where ρ is the density, u and v are the velocity components in x and y directions, p is the pressure, and E is the total energy. The pressure is related to the total energy by

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

with a constant ratio of specific heats $\gamma = 1.4$ for air. Define a flux vector with two components, i.e., $F = (f, g)$. Eq. (1a) can be expressed in the following divergence form

$$\frac{\partial Q}{\partial t} + \nabla \bullet F = 0\ . \tag{1d}$$



(a) 1st order **(b) 2nd order** **(c) 3rd order**

Figure 1. Placement of solution (●) and flux (🟩) points for a triangular element

Eq. (1) is to be solved on a non-overlapping simplex grid with proper initial and boundary conditions. Within each cell or element, we define two different sets of grid points, i.e., the *solution points* and *flux points*. The solution points are the locations where the nodal values of the conservative variables Q are specified (usually Gauss quadrature points). Flux points are the locations where the nodal values of fluxes F are computed. The solution unknowns or degrees of freedom in the SD method are the conservative variables at the solution points. Figure 1 displays possible placements of solution and flux points for the first to third-order SD schemes. Let the position vector of the *j-th* solution point at cell i be denoted by $\mathbf{r}_{j,i}$, and the *k-th* flux point at cell i be denoted by $\mathbf{r}_{k,i}$. Denote $Q_{j,i}$ the solution at $\mathbf{r}_{j,i}$. Given the solutions at $\mathbf{r}_{j,i}$, an element-wise degree p polynomial can be constructed using Lagrange-type polynomial basis, i.e.,

$$Q_i(\mathbf{r}) = \sum_{j=1}^{N_p} L_{j,i}(\mathbf{r}) Q_{j,i}\ , \tag{2}$$

where $L_{j,i}(\mathbf{r})$ are the cardinal basis functions and N_p is the number of basis functions required to support a degree p polynomial reconstruction. Obviously, the locations of the solution $\mathbf{r}_{j,i}$ uniquely determine the cardinal basis functions $L_{j,i}(\mathbf{r})$. With the polynomial distribution given in (2), the solutions of Q at the flux points $\mathbf{r}_{k,i}$ can be computed easily from

$$Q(\mathbf{r}_{k,i}) = \sum_{j=1}^{N_p} L_{j,i}(\mathbf{r}_{k,i}) Q_{j,i}\ . \tag{3}$$

Since the solutions are element-wise polynomials, they are discontinuous across element boundaries. As a result, the fluxes at the element interfaces are not uniquely defined, for example at the corner and face points shown in Figure 2. At the corner point, five solutions exist from all the cells (*A, B, C, D* and *E*) sharing the point. At the face point between cells C and D, two solutions exist. The naïve approach is to compute an averaged solution from these multiple solutions and then determine the flux based on the averaged solution. However it is well known that this naïve approach is equivalent to central differencing and is not stable. An alternative approach is to find the physical solution Q at the corner and face points at time $t = 0^+$ with the discontinuous solutions as the initial condition for the Euler equations at $t = 0$. This idea is of course due to Godunov[19], who pioneered the well-known Godunov-type finite volume methods, which become the standard method for conservation laws[20]. Unfortunately, this “multi-dimensional Riemann problem” shown in Figure 2 is very difficult to solve, either analytically or numerically.

Simpler approximate Riemann solvers must be found to determine these fluxes. We again turn to the FV method to look for inspirations. Obviously, in order to ensure conservation, the normal component of the flux vector on each face should be identical for the two cells sharing the face. Physically this means a mass flux going out of a cell must completely enter the neighboring cell without mass generation or loss. To ensure conservation, a one dimensional Riemann solver is employed in the face normal direction to compute the common normal flux. Consider the face flux point shown in Figure 3, and denote the outgoing normal from cell *C* to cell 1 ***n₁***. For this interface point, Q_L is computed from cell C and Q_R is computed from cell 1. Then the common normal component of the flux can be computed with any Riemann solvers such as the Rusanov[11] or Roe[12] flux. In the case of the simpler Rusanov flux, the normal component is computed from

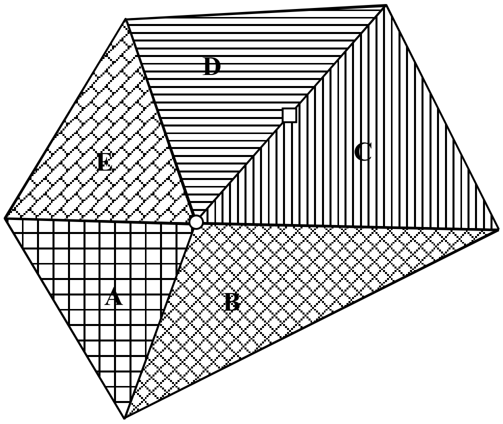


Figure 2. Illustration of multi-dimensional Riemann problems at the corner and face point

$$F_n = F_n(Q_L, Q_R, \mathbf{n}) = \frac{1}{2}\left\{[F(Q_L) + F(Q_R)] \bullet \mathbf{n} - (\bar{v}_n + \bar{c})(Q_R - Q_L)\right\}, \tag{4}$$

where $\bar{v}_n$ is the average normal velocity and $\bar{c}$ is the average speed of sound computed from the left and right solutions. Since the tangential component of the flux does not affect the conservation property, we have the complete freedom in determining it at the face point. In fact, it is not strictly necessary to have a unique tangential component physically at the face point (e.g. think of a contact discontinuity in which density is discontinuous). Let the unit vector in the tangential direction be ***l***. Here we offer two possibilities. One is to use a unique tangential component by averaging the two tangential components from both sides of the face, i.e.,

$$F_l = F_l(Q_L, Q_R, \mathbf{l}) = \frac{1}{2}\left\{[F(Q_L) + F(Q_R)] \bullet \mathbf{l}\right\}. \tag{5}$$

The other possibility is to use its own tangential component from the current cell, allowing the tangential component to be discontinuous. Therefore, the tangential component of the flux on either side is not modified. For the left cell, the tangential and normal component are $((F(Q_L) \bullet \mathbf{l}, F_n)$, and for the right cell, they become $(F(Q_R) \bullet \mathbf{l}, F_n)$.

For a corner flux point in cell *C*, two faces (from cell *C*) share the corner point, as shown in Figure 3. Let the unit normals of the two faces be $\mathbf{n}_1$ and $\mathbf{n}_2$. Once again, the normal components of flux F_{n1} and F_{n2} in $\mathbf{n}_1$ and $\mathbf{n}_2$ directions are computed with a one-dimensional Riemann solver in the normal directions. The full flux vector can then be uniquely determined from the two normal flux components

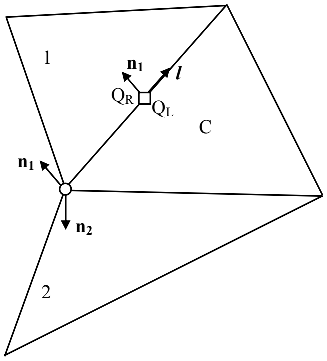


Figure 3. Flux computation for a corner (○) and a face (□) point using one-dimensional Riemann solvers.

$$F \bullet \mathbf{n}_1 = F_{n1}, \tag{6a}$$

$$F \bullet \mathbf{n}_2 = F_{n2}. \tag{6b}$$

It is important to emphasize here that fluxes at cell corner points do not have unique values for all the cells sharing the corner. In spite of that, local conservation is guaranteed because neighboring cells do share a common normal flux at all the flux points. Once the fluxes at all the flux points are re-computed, they are used to form a degree *p+1* polynomial, i.e.,

$$F_i(\mathbf{r}) = \sum_{k=1}^{N_{p+1}} M_{k,i}(\mathbf{r}) F_{k,i} \,, \tag{7}$$

where $M_{k,i}(\mathbf{r})$ are the set of cardinal basis functions defined by $\mathbf{r}_{k,i}$ and $F_{k,i} = F(\mathbf{r}_{k,i})$. Obviously, the divergence of the flux at any point within the cell can be computed using

$$\nabla \bullet F_i(\mathbf{r}) = \sum_{k=1}^{N_{p+1}} \nabla M_{k,i}(\mathbf{r}) \bullet F_{k,i} \,. \tag{8}$$

To update the solutions at the solution points $\mathbf{r}_{j,i}$, we need to evaluate the divergence at these points, which can be easily computed according to

$$\nabla \bullet F_i(\mathbf{r}_{j,i}) = \sum_{k=1}^{N_{p+1}} \nabla M_{k,i}(\mathbf{r}_{j,i}) \bullet F_{k,i} \,. \tag{9}$$

Finally the semi-discrete scheme to update the solution unknowns can be written as

$$\frac{dQ_{j,i}}{dt} + \sum_{k=1}^{N_{p+1}} \nabla M_{k,i}(\mathbf{r}_{j,i}) \bullet F_{k,i} = 0 \,. \tag{10}$$

For time integration, high-order TVD (or SSP) Runge-Kutta schemes[21,22] are employed.

III. Efficient Implementation and Conservation Property

In the last section, we avoided the implementation and conservation issues in order to focus on presenting the basic idea of the SD method. The reconstruction formulas for the solution and flux presented in (2), (7) may give the readers the impression that each cell has a different set of reconstruction coefficients. It will be shown that for triangles with straight edges (most of the cells except curved wall boundary cells), the reconstruction coefficients are universal for all triangles. This is true because any triangle can be transformed to a standard triangle as shown in Figure 4 through the following linear transformation:

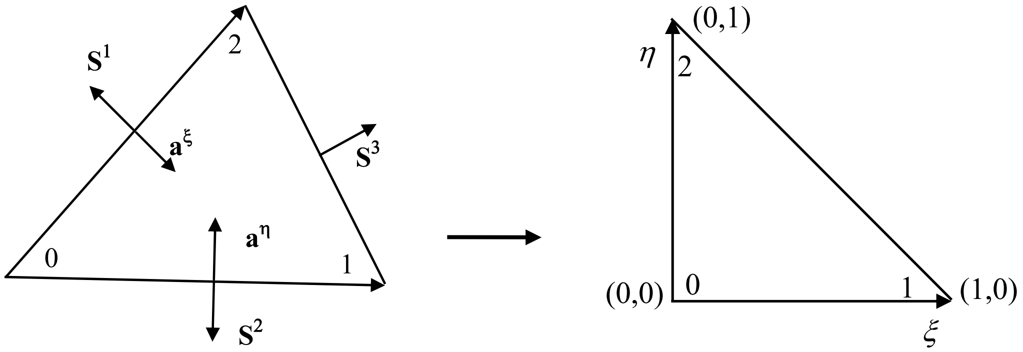


Figure 4. Transformation from a physical element to the standard element

$$\mathbf{r} = \mathbf{r}_{0,i} + \xi(\mathbf{r}_{1,i} - \mathbf{r}_{0,i}) + \eta(\mathbf{r}_{2,i} - \mathbf{r}_{0,i}),\ 0 \le \xi, \eta \le 1 \quad \text{and} \quad \xi + \eta \le 1 \tag{11}$$

where $\mathbf{r}_{0,i}$, $\mathbf{r}_{1,i}$, and $\mathbf{r}_{2,i}$ are the three vertices of cell *i*. If the solution points $\mathbf{r}_{j,i}$ and flux points $\mathbf{r}_{k,i}$ are distributed in a geometrically similar manner for all cells, they all have the same local position $\boldsymbol{\xi}_j$ and $\boldsymbol{\xi}_k$ with $\boldsymbol{\xi} = (\xi, \eta)$. It is shown in Reference 2 that a universal reconstruction can be written as

$$Q_i(\boldsymbol{\xi}) = \sum_{j=1}^{N_p} L_j(\boldsymbol{\xi}) Q_{j,i}\ , \tag{12}$$

where the cardinal basis functions $L_j(\boldsymbol{\xi})$ are universal for all triangles. Therefore the solution at the flux points can be computed using

$$Q_{k,i} \equiv Q_i(\boldsymbol{\xi}_k) = \sum_{j=1}^{N_p} l_{kj} Q_{j,i}\ , \tag{13}$$

where $l_{kj} = L_j(\boldsymbol{\xi}_k)$. Similarly the reconstruction polynomial for the flux can be written as

$$F_i(\boldsymbol{\xi}) = \sum_{k=1}^{N_{p+1}} M_k(\boldsymbol{\xi}) F_{k,i}\ , \tag{14}$$

where $\{M_k(\boldsymbol{\xi})\}$ are the universal cardinal basis functions based on the flux points. It is then straightforward to show that the gradient of *F* on the computational domain (standard element) takes the following universal form for all cells

$$\nabla F_i(\boldsymbol{\xi}) = \sum_{k=1}^{N_{p+1}} \nabla M_k(\boldsymbol{\xi}) F_{k,i}\ . \tag{15}$$

From (15), the gradients of the flux at the solution points can be computed according to

$$\frac{\partial F_i(\boldsymbol{\xi}_j)}{\partial \xi} = \sum_{k=1}^{N_{p+1}} m_{jk,\xi} F_{k,i}\ . \tag{16}$$

$$\frac{\partial F_i(\boldsymbol{\xi}_j)}{\partial \eta} = \sum_{k=1}^{N_{p+1}} m_{jk,\eta} F_{k,i}\ . \tag{17}$$

where $m_{jk,\xi} = \dfrac{\partial M_k(\boldsymbol{\xi}_j)}{\partial \xi}$ and $m_{jk,\eta} = \dfrac{\partial M_k(\boldsymbol{\xi}_j)}{\partial \eta}$. By applying the chain rule, we can easily relate the divergence of *F* in the physical domain to the gradients of the flux in the computational domain

$$\nabla \bullet F_i(\mathbf{r}) = \frac{\partial f_i(\mathbf{r})}{\partial x} + \frac{\partial g_i(\mathbf{r})}{\partial y} = \frac{\partial f_i(\boldsymbol{\xi})}{\partial \xi} \xi_{x,i} + \frac{\partial f_i(\boldsymbol{\xi})}{\partial \eta} \eta_{x,i} + \frac{\partial g_i(\boldsymbol{\xi})}{\partial \xi} \xi_{y,i} + \frac{\partial g_i(\boldsymbol{\xi})}{\partial \eta} \eta_{y,i}\ . \tag{18}$$

For the linear transformation given in (11), it is easy to show that

$$\begin{bmatrix} \xi_{x,i} & \xi_{y,i} \\ \eta_{x,i} & \eta_{y,i} \end{bmatrix} = \frac{1}{2V_i} \begin{bmatrix} y_{2,i} - y_{0,i} & -x_{2,i} + x_{0,i} \\ -y_{1,i} + y_{0,i} & x_{1,i} - x_{0,i} \end{bmatrix}, \tag{19}$$

where V_i is the volume of cell *i*. Let $\mathbf{a}^{\xi}$ and $\mathbf{a}^{\eta}$ be the inward-pointing area vectors of face 02 and face 01 in Figure 4. Obviously, we have

$$\begin{aligned} (\xi_{x,i}, \xi_{y,i}) &= \frac{\mathbf{a}_i^{\xi}}{2V_i}, \\ (\eta_{x,i}, \eta_{y,i}) &= \frac{\mathbf{a}_i^{\eta}}{2V_i}. \end{aligned} \tag{20}$$

Eq. (18) can be more concisely written as

$$\nabla \bullet F_i(\mathbf{r}) = \frac{1}{2V_i} \left[\frac{\partial F_i(\boldsymbol{\xi})}{\partial \xi} \bullet \mathbf{a}_i^{\xi} + \frac{\partial F_i(\boldsymbol{\xi})}{\partial \eta} \bullet \mathbf{a}_i^{\eta} \right]. \tag{21}$$

Finally (10) becomes

$$\begin{aligned} &\frac{dQ_{j,i}}{dt} + \frac{1}{2V_i} \left[\frac{\partial F_i(\boldsymbol{\xi}_j)}{\partial \xi} \bullet \mathbf{a}_i^{\xi} + \frac{\partial F_i(\boldsymbol{\xi}_j)}{\partial \eta} \bullet \mathbf{a}_i^{\eta} \right] = \\ &\frac{dQ_{j,i}}{dt} + \frac{1}{2V_i} \left[\mathbf{a}_i^{\xi} \bullet \sum_{k=1}^{N_{p+1}} m_{jk,\xi} F_{k,i} + \mathbf{a}_i^{\eta} \bullet \sum_{k=1}^{N_{p+1}} m_{jk,\eta} F_{k,i} \right] = 0. \end{aligned} \tag{22}$$

Note that for all triangles (with straight faces), only the two vectors $\frac{\mathbf{a}_i^{\xi}}{2V_i}, \frac{\mathbf{a}_i^{\eta}}{2V_i}$ need to be stores. The computation of the flux gradients on the computational domain is universal for all triangles. Obviously this formulation is much simpler than that of the DG and SV methods, and is much easier to implement too.

In order to prove conservation, we need to show that the integral form of (1) is satisfied in each cell, i.e.,

$$\int_{V_i} \left(\frac{\partial Q}{\partial t} + \nabla \bullet F \right) dV = \frac{d}{dt} \int_{V_i} Q dV + \oint_{\partial V_i} F \bullet \mathbf{n} \, dS = 0 . \tag{23}$$

The volume and surface integrals are computed using quadrature formulas based on the solutions at the solution points and fluxes at the flux points

$$\int_{V_i} Q dV = V_i \sum_{j=1}^{N_p} w_j Q_{j,i} \tag{24}$$

$$\oint_{\partial V_i} F \bullet \mathbf{n} \, dS = \sum_{l=1}^{3} \left(\mathbf{n}_l \bullet \int_{S_i^l} F \, dS \right) = \sum_{l=1}^{3} \left(\mathbf{S}_i^l \bullet \sum_{k=1}^{N_{p+1}} w_k^l F_{k,i} \right), \tag{25}$$

where w_j is the volume integral quadrature weights, and w_k^l are the surface integral quadrature weights for face *l*, and $\mathbf{S}_i^l$ is the outward area vector of face *l*. Using the fact that the area vectors form a closed surface, $\sum_{l=1}^{3} \mathbf{S}_i^l = 0$, or $\mathbf{S}_i^3 = -\mathbf{S}_i^1 - \mathbf{S}_i^2$, and $\mathbf{S}_i^1 = -\mathbf{a}^\xi$, $\mathbf{S}_i^2 = -\mathbf{a}^\eta$, (25) can be further written as

$$\oint_{\partial V_i} F \bullet \mathbf{n}\, dS = -\left(\mathbf{a}_i^\xi \bullet \sum_{k=1}^{N_{p+1}} w_k^\xi F_{k,i} + \mathbf{a}_i^\eta \bullet \sum_{k=1}^{N_{p+1}} w_k^\eta F_{k,i} \right). \tag{26}$$

where $w_k^\xi = w_k^1 - w_k^3$, and $w_k^\eta = w_k^2 - w_k^3$. Obviously, only flux points at element interfaces are used in the surface integral. Therefore, for all interior flux points, the weights should be zero. Substituting (22) into (24), we obtain

$$\begin{aligned} \frac{d}{dt}\int_{V_i} Q dV &= V_i \sum_{j=1}^{N_p} w_j \frac{dQ_{j,i}}{dt} = -\frac{1}{2}\sum_{j=1}^{N_p} w_j \sum_{k=1}^{N_{p+1}} F_{k,i} \bullet (m_{jk,\xi}\mathbf{a}_i^\xi + m_{jk,\eta}\mathbf{a}_i^\eta) \\ &= -\left(\mathbf{a}_i^\xi \bullet \frac{1}{2} \sum_{k=1}^{N_{p+1}} \sum_{j=1}^{N_p} w_j m_{jk,\xi} F_{k,i} + \mathbf{a}_i^\eta \bullet \frac{1}{2} \sum_{k=1}^{N_{p+1}} \sum_{j=1}^{N_p} w_j m_{jk,\eta} F_{k,i} \right). \end{aligned} \tag{27}$$

Comparing (26) and (27), the conservation conditions are

$$w_k^\xi = -\frac{1}{2}\sum_{j=1}^{N_p} w_j m_{jk,\xi}, \text{ and } w_k^\eta = -\frac{1}{2}\sum_{j=1}^{N_p} w_j m_{jk,\eta}. \tag{28}$$

Since these equations depend only on the locations of the solution and quadrature points, they can be satisfied by properly placing the solution and flux points. In face, these conditions are satisfied for all the placements shown in Figure 1.

IV. Numerical Results

A. Accuracy Study with Vortex Evolution Problem

This is an idealized problem for the Euler equations in 2D used by Shu[23]. The mean flow is $\{\rho, u, v, p\} = \{1, 1, 1, 1\}$. An isotropic vortex is then added to the mean flow, i.e., with perturbations in *u*, *v*, and temperature $T = p/\rho$, and no perturbation in entropy $S = p/\rho^\gamma$:

$$(\delta u, \delta v) = \frac{\varepsilon}{2\pi} e^{0.5(1-r^2)}(-\bar{y}, \bar{x}),$$

$$\delta T = -\frac{(\gamma-1)\varepsilon^2}{8\gamma\pi^2} e^{1-r^2},$$

$$\delta S = 0,$$

where $r^2 = \bar{x}^2 + \bar{y}^2$, $\bar{x} = x - 5$, $\bar{y} = y - 5$, and the vortex strength $\varepsilon = 5$. If the computational domain is infinitely big, the exact solution of the Euler equations with the above initial conditions is just the passive convection of the isotropic vortex with the mean velocity (1, 1). In the following accuracy study, the computational domain is taken to be [0, 10]x[0, 10], with characteristic inflow and outflow boundary conditions imposed on the boundaries.

The numerical simulations were carried out until t = 2 on a set of irregular meshes. The coarsest mesh is shown in Figure 5. The finer meshes are generated recursively by cutting each coarser grid cell into

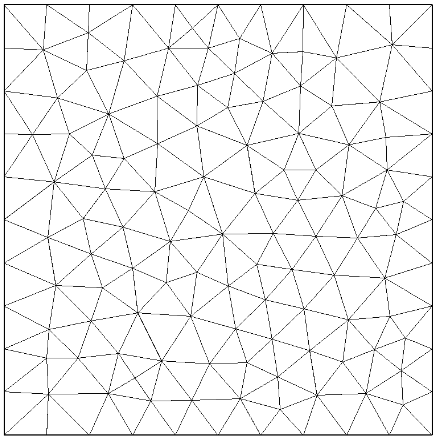
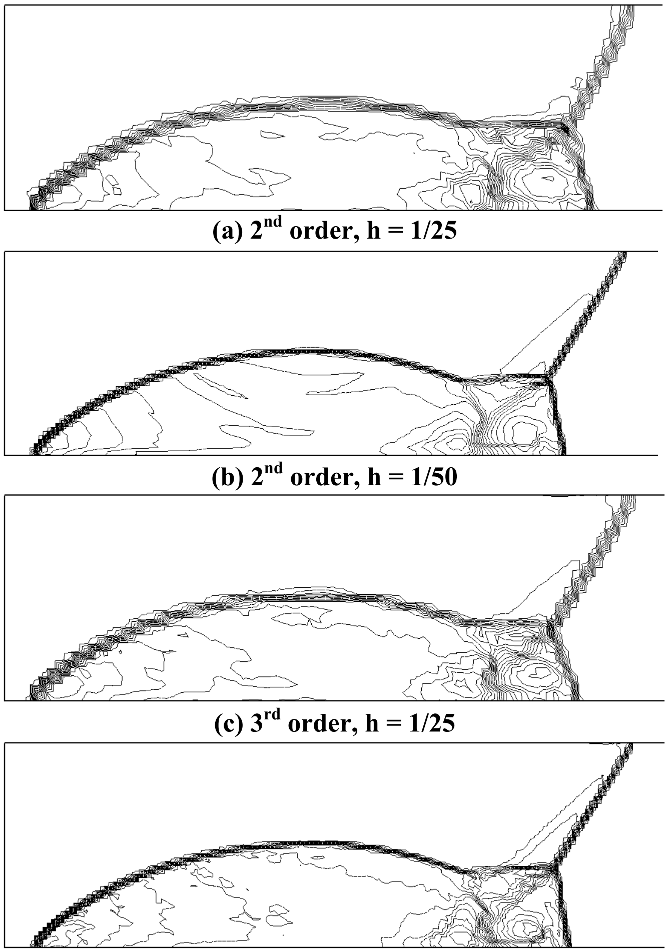


Figure 5. Irregular "10x10x2" Grid

four finer grid cells. The Rusanov flux was used in the simulations. The L_1 and L_∞ norms in density are presented for SD schemes of second and third order in Table 1. The errors presented in the tables were made time step independent by using sufficiently small time steps. Note that all the simulations have reached the desired order of accuracy in the L_1 and L_∞ norms. The SD method is about 25% faster than the SV method for this case.

B. Double Mach Reflection

This problem is also a standard test case for high-order methods[24] and has been studied extensively by many researchers. The computational domain for this problem is chosen to be [0, 4] x [0, 1]. The reflecting wall lies at the bottom of the computational domain starting from x=1/6. Initially a right-moving Mach 10 shock is positioned at x=1/6, y=0 and makes a 60° angle with the x-axis. For the bottom boundary, the exact post-shock condition is imposed for the region from x=0 to x=1/6 and a solid wall boundary condition is used for the rest. For the top boundary of the computational domain, the solution is set to describe the exact motion of the Mach 10 shock. The left boundary is set as the exact post-shock condition, while the right boundary is set as outflow boundary. Two triangular grids were generated with approximate mesh sizes of 1/25, 1/50. These meshes have 5,000, 18,800 triangular cells respectively, corresponding to 15,000, 56,400, unknowns for the 2nd-order SD scheme, and 30,000, 112,800 unknowns for the 3rd-order SD scheme. All the simulations were carried until t=0.2 using the Roe flux and TVD limiter similar to the one developed for the SV method[9]. Figure 6 shows the density contours computed with the second and third-order SD scheme on the coarse, and fine grids. It is obvious that the third order SD scheme has higher resolution than the second-order SD scheme for the complex flow structures near the double Mach stem. This case demonstrates that the SD method can be used for shock capturing, and satisfactory results have been obtained for this case.


(a) 2nd order, h = 1/25

(b) 2nd order, h = 1/50

(c) 3rd order, h = 1/25

(d) 3rd order, h = 1/50

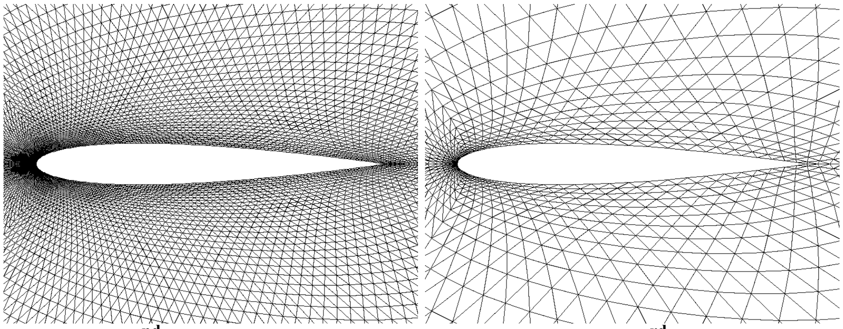
Figure 6. Density contours for the double Mach reflection problem

C. Subsonic Flow over a NACA0012 Airfoil

As a final demonstration for a more realistic geometry, subsonic flow around a NACA0012 airfoil at Mach = 0.4, and angle of attack of 5 degrees is simulated. In this simulation, the computational results using the 3rd order SD scheme on a coarse mesh with 72x24x2 triangles are compared with those using a 2nd order MUSCL type FV method[25] on a much finer mesh with 192x64x2 triangles. Therefore the number of DOFs in the FV simulation is 24,576 while it is 20,736 in the SD simulation. The entropy production in the solution is used as the indicator for the solution accuracy. For the 3rd-order SD scheme, the boundary is approximated with 72 piece-wise quadratic segments. For the 2nd order FV scheme, the airfoil surface is approximated with 192 linear segments. The computational meshes used for both the SD and FV methods are displayed in Figure 7. The outer boundary is 20 chords away from the center of the airfoil. The computed Mach contours computed with both the SD and FV schemes are plotted in Figures 8. Note that the agreement is very good. The average entropy error with the 2nd order FV method is 1.04e-5, while the average entropy error with the 3rd order SD scheme is 4.86e-6, which is more than a factor of 2 smaller. The entropy errors along the airfoil surface are plotted for both computational results in Figure 9. Note that although the 2nd-order FV scheme used a much finer grid, the solution quality of the 3rd order SD scheme is superior.

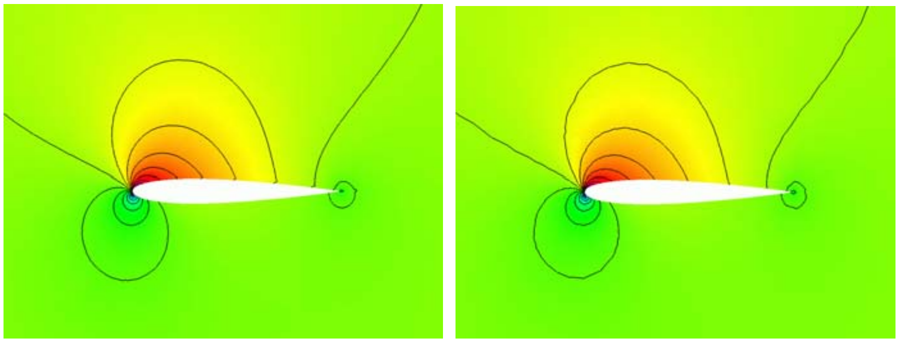
V. Conclusions

In this paper, the spectral difference method has been successfully extended to Euler equations on unstructured grids. The method combines the best features of structured and unstructured grid methods in which the structured distribution of discrete variables in each unstructured cell maintains computational efficiency and geometric flexibility. It utilizes the concept of discontinuous and high-order local representations to achieve conservation and high accuracy. Universal reconstructions are obtained by distributing unknown and flux points in a geometrically similar manner for all unstructured cells. The flux derivatives needed to update the conservative unknowns are expressed as universal weighted sums of the fluxes, leading to great computational efficiency. An important aspect of the method is that the number of Riemann solvers per unknown decreases as the order of accuracy increases, reducing the cost for higher order. Placements of the unknown and flux points with various orders of accuracy are given for triangular elements. Accuracy studies of the method are carried out with the vortex propagation problem and the order of accuracy is numerically verified. A monotonicity solution limiter has been implemented for discontinuity capturing, successfully employed to simulate a double Mach reflection problem. The method is also applied to high-order boundary representations, and satisfactory results have been obtained for a subsonic flow around the NACA0012 airfoil. The 3rd-order SD was shown to produce more accurate results than a 2nd-order FV method on a much coarser grid with fewer solution unknowns. Future research areas in the SD method include extension to even higher order of accuracy, and to Navier-Stokes equations and three dimensions.



(a) 2nd FV grid (b) 3rd SD grid

Figure 7. Computational grids for subsonic flow around a NACA0012 airfoil



(a) 2nd order FV (b) 3rd order SD

Figure 8. Comparison of Mach contours between the FV and SD methods

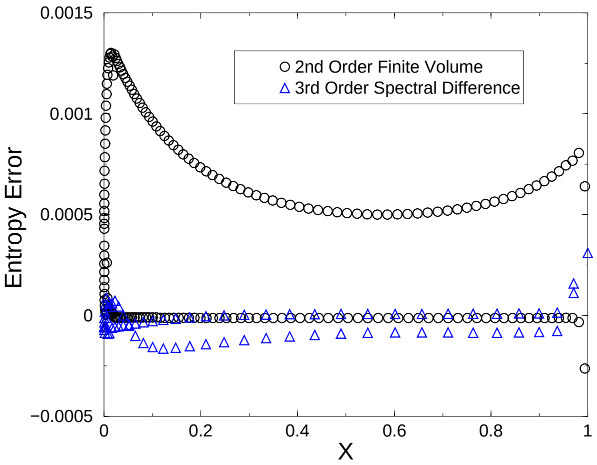


Figure 9. Comparison of entropy error along the airfoil surface

References

[1]Y. Liu, M. Vinokur, Z.J. Wang, "Discontinuous Spectral Difference Method for Conservation Laws on Unstructured Grids," in Proceedings of the 3rd International Conference in CFD, Toronto, Canada July 2004.

[2]Y. Liu, M. Vinokur, Z.J. Wang, "Spectral Difference Method for Unstructured Grids I: Basic Formulation," submitted.

[3]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 52, 411-435 (1989).

[4]Cockburn and C.-W. Shu, The Runge-Kutta discontinuous Garlerkin method for conservation laws V: multidimensional systems, J. Comput. Phys., 141, 199 - 224, (1998).

[5] F. Bassi and S. Rebay, High-order accurate discontinuous finite element solution of the 2D Euler equations, J. Comput. Phys. 138, 251-285 (1997).

[6] Z.J. Wang, Spectral (finite) volume method for conservation laws on unstructured grids: I. Basic formulation, J. Comput. Phys. 178, 210 (2002).

[7] Z.J. Wang and Y. Liu, Spectral (finite) volume method for conservation laws on unstructured grids II: extension to two-dimensional scalar equation, J. Comput. Phys. 179, 665-697 (2002).

[8] Z.J. Wang and Y. Liu, Spectral (finite) volume method for conservation laws on unstructured grids III: one-dimensional systems and partition optimization, J. Scientific Comput. 20, 137-157 (2004).

[9] Z.J. Wang, L. Zhang, and Y. Liu and, Spectral (finite) volume method for conservation laws on unstructured grids IV: extension to two-dimensional systems, J. Comput. Phys. 194, 716-741 (2004).

[10] Y. Liu, M. Vinokur, and Z.J. Wang, Spectral (finite) volume method for conservation laws on unstructured grids V: extension to three-dimensional systems, submitted to J. Comput. Phys. 2005.

[11] V.V. Rusanov, Calculation of interaction of non-steady shock waves with obstacles, J. Comput. Math. Phys. USSR 1, 267-279 (1961).

[12] P.L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, J. Comput. Phys. 43 357-372 (1981).

[13] D. Sridar and N. Balakrishnan, An upwind finite difference scheme for meshless solvers, J. Comput. Phys. 189 1-29 (2003).

[14] J. Barth and P.O. Frederickson, High-order solution of the Euler equations on unstructured grids using quadratic reconstruction, AIAA Paper No. 90-0013 (1990).

[15] M. Delanaye and Y. Liu, Quadratic reconstruction finite volume schemes on 3D arbitrary unstructured polyhedral grids, AIAA Paper No. 99-3259-CP, 1999.

[16] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, Spectral Methods in Fluid Dynamics (Springer-Verlag, New York, 1987).

[17] D.A. Kopriva, A conservative staggered-grid Chebyshev multidomain method for compressible flows. II semi-structured method, J. Comput. Phys. 128, 475 (1996).

[18] D.A. Kopriva, A staggered-grid multidomain spectral method for the compressible Navier–Stokes equations, J. Comput. Phys., 143, No. 1, 125 (1998).

[19] S.K. Godunov, A finite-difference method for the numerical computation of discontinuous solutions of the equations of fluid dynamics, Mat. Sb. 47, 271 (1959).

[20] B. van Leer, Towards the ultimate conservative difference scheme V. a second order sequel to Godunov's method, J. Comput. Phys. 32, 101-136 (1979).

[21] C.-W. Shu, Total-variation-diminishing time discretizations, SIAM J. Sci. Stat. Comput. 9, 1073 (1988).

[22] R. J. Spiteri and S. J. Ruuth, A new class of optimal high-order strong-stability-preserving time discretization methods, SIAM J. Numer. Anal., 40 (2002), pp. 469–491.

[23] C.-W. Shu, Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws, in Advanced Numerical Approximation of Nonlinear Hyperbolic Equations, B. Cockburn, C. Johnson, C.-W. Shu and E. Tadmor (Editor: A. Quarteroni), Lecture Notes in Mathematics, volume 1697, Springer, 1998, pp.325-432.

[24] P. Woodward and P. Colella, The numerical simulation of two-dimensional fluid flow with strong shocks, J. Comput. Phys., 54, 115-173 (1984).

[25] Z.J. Wang, A fast nested multi-grid viscous flow solver for adaptive Cartesian/quad grids, International Journal for Numerical Methods in Fluids, 33, 657-680 (2000).

Table I. Accuracy on the vortex propagation case (t = 2) (irregular grids), Rusanov flux

Order of Accuracy	*Grid*	L_1 *error*	L_1 *order*	L_∞ *error*	L_∞ *order*
2	10x10x2	6.82e-3	-	1.11e-1	-
	20x20x2	1.72e-3	1.99	3.66e-2	1.62
	40x40x2	4.75e-4	1.86	8.99e-3	2.03
	80x80x2	1.22e-4	1.96	2.22e-3	2.02
	160x160x2	3.07e-5	1.99	6.32e-4	1.81
3	10x10x2	1.94e-3	-	5.43e-2	-
	20x20x2	3.55e-4	2.45	7.50e-3	2.86
	40x40x2	5.97e-5	2.57	1.32e-3	2.51
	80x80x2	9.79e-6	2.61	1.94e-4	2.77
	160x160x2	1.52e-6	2.69	3.22e-5	2.59