Journal of Computational Physics 239 (2013) 138–146

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Short note

A comparison of computational efficiencies of spectral difference method and correction procedure via reconstruction

Chunlei Liang *, Christopher Cox, Michael Plesniak

Department of Mechanical and Aerospace Engineering, George Washington University, DC 20052, United States

ARTICLE INFO

Article history:
Received 18 May 2012
Received in revised form 14 December 2012
Accepted 3 January 2013
Available online 21 January 2013

Keywords:
Spectral difference method
Correction procedure via reconstruction
Unstructured grid
Quadrilateral element

ABSTRACT

We report computational efficiencies of two types of numerical solvers. The first type uses the spectral difference (SD) method and the second one uses the correction procedure via reconstruction (CPR). In this paper, we employ the lumped g_2 scheme proposed by Huynh as an example of the CPR approach. The solvers deal with both inviscid Euler equations and Navier–Stokes equations on 2D unstructured grids which are comprised of all quadrilateral cells. Both types of solvers are programmed using Fortran 90 with similar management of data structures. We employ identical time marching schemes for both SD and CPR methods. Spatial 3rd and 4th order of accuracy for both methods is demonstrated by a study of a moving inviscid vortex. The comparisons were directed to measure the computational efficiency of both SD and CPR methods in spatial discretization. With respect to the fourth order methods, CPR is 27% faster than SD for inviscid flow, and more promisingly, CPR is over 40% faster than SD for viscous flow.

1. Introduction

The spectral difference (SD) method is a high-order efficient approach based on differential strong form of the governing equations. The SD method combines elements from finite-volume and finite-difference techniques. It is able to achieve optimal order of accuracy for various compressible flows [6–8,16]. The method is particularly attractive because it is conservative, and has a simple formulation and is easy to implement. The first papers of unstructured SD method [10,21] were developed for wave equations and compressible Euler equations. Recently, it has been developed for solving 3D turbulent compressible flows [9,13]. It should be mentioned, however, that the SD method often times approximates the weak solution of the governing equations, and that this type of scheme is equivalent to a quadrature-free discontinuous galerkin (DG) method for nonlinear hyperbolic conservation laws under certain conditions [12].

In 2007, [3] introduced a new approach to high-order accuracy by solving the equations again in differential form. The approach, originally called flux reconstruction (FR), results in numerous schemes with favorable properties. The framework was applied to solve diffusion problems using quadrilateral meshes in [4]. In 2009, [19] extended the FR idea to 2D triangular and quadrilateral mixed meshes with the lifting collocation penalty (LCP) framework. The involved authors later combined the names of "FR" and "LCP" and called them correction procedure via reconstruction (CPR). The CPR formulation is believed to be among the most efficient discontinuous methods in terms of the number of operations in [20].

Correction functions are used to correct the discontinuous flux function within an element in order to insure flux continuity across element interfaces. The correction function proposed by Huynh [4] and implemented here is expressed in terms of a combination of Radau polynomials, where the zeros of the derivative of the correction function coincide with

* Corresponding author. Tel.: +1 202 994 7073.
E-mail address: chliang@gwu.edu (C. Liang).

0021-9991/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2013.01.001

the Legendre–Lobatto points. This fact lends itself to choosing the Lobatto points as the solution points such that the jump in flux across an interface is only due to a correction to the discontinuous flux function at the interface, without involving corrections to interior flux points. Recently, [18] reported a stability proof using a flux reconstruction formulation with energy estimates for a one parameter family of schemes where the family was expressed in terms of the Legendre polynomials instead of the Radau polynomials, proving stability for all orders of accuracy on unstructured grids.

Although we conceptually expect that the CPR method should be faster than the SD method, a direct comparison of CPR and SD solvers using quadrilateral elements is not yet available. Recently, an alternate form of the SD method using Raviart–Thomas spaces has been proven stable for triangular elements in [1], which paves the way for further comparisons of the CPR and SD schemes.

The following section presents the mathematical formulation of the governing equations in both physical and computational domains. We briefly review the SD method in Section 3. Section 4 provides the numerical formulation of the CPR method. Section 5 demonstrates the order of accuracy of our CPR and SD solvers. Section 6 presents our solution of inviscid and viscous subsonic flows around a NACA0012 airfoil. In Section 7, CPU processing time is reported for both methods. Section 8 concludes this paper.

2. Mathematical formulation

We consider the 2D compressible Navier–Stokes equations, for which the conservative form is given by

$$\frac{\partial \mathbf{Q}}{\partial t}+\frac{\partial \mathbf{F}}{\partial x}+\frac{\partial \mathbf{G}}{\partial y}=0, \tag{1}$$

where $\mathbf{Q}$ is the vector of conservative variables

$$\mathbf{Q}=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ E\end{bmatrix}, \tag{2}$$

and $\mathbf{F}$ and $\mathbf{G}$ are the flux vectors in the x and y directions, respectively, which can be decomposed into inviscid and viscous components as

$$\mathbf{F}=\mathbf{F}_I(Q)-\mathbf{F}_V(Q,\nabla Q), \tag{3}$$

$$\mathbf{G}=\mathbf{G}_I(Q)-\mathbf{G}_V(Q,\nabla Q), \tag{4}$$

where the inviscid flux vectors $\mathbf{F}_I(Q)$ and $\mathbf{G}_I(Q)$ are

$$\mathbf{F}_I(Q)=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ (E+p)u\end{bmatrix}, \tag{5}$$

$$\mathbf{G}_I(Q)=\begin{bmatrix}\rho v\\ \rho vu\\ \rho v^2+p\\ (E+p)v\end{bmatrix}, \tag{6}$$

and the viscous flux vectors $\mathbf{F}_V(Q,\nabla Q)$ and $\mathbf{G}_V(Q,\nabla Q)$ are

$$\mathbf{F}_V(Q,\nabla Q)=\begin{bmatrix}0\\ \tau_{xx}\\ \tau_{xy}\\ k\frac{\partial T}{\partial x}+u\tau_{xx}+v\tau_{xy}\end{bmatrix}, \tag{7}$$

$$\mathbf{G}_V(Q,\nabla Q)=\begin{bmatrix}0\\ \tau_{yx}\\ \tau_{yy}\\ k\frac{\partial T}{\partial y}+u\tau_{yx}+v\tau_{yy}\end{bmatrix}. \tag{8}$$

The total energy E is

$$E=\frac{p}{(\gamma-1)}+\frac{1}{2}\rho(u^2+v^2). \tag{9}$$

In the governing equations presented above, u and v are the components of velocity in the x and y directions and ρ, p, T, γ and k represent the density, pressure, temperature, specific heat ratio and thermal conductivity, respectively. The components of shear stress τ_{ij} that appear in the viscous flux vectors are those obtained from Stokes' hypothesis.

Considering the non-uniformity of the physical grid (x, y), we employ an iso-parametric mapping to transform the fully-conservative equations in order to solve them over the computational domain $((x, y) \rightarrow (\xi, \eta))$, which allows for universal reconstruction via polynomials in both the CPR and SD method. The computational domain is represented by standard square elements ($0 \leqslant \xi \leqslant 1$ and $0 \leqslant \eta \leqslant 1$) over which the governing equations can be efficiently solved. The transformed equations take the following form:

$$\frac{\partial \tilde{\mathbf{Q}}}{\partial t} + \frac{\partial \tilde{\mathbf{F}}}{\partial \xi} + \frac{\partial \tilde{\mathbf{G}}}{\partial \eta} = 0, \tag{10}$$

where

$$\tilde{\mathbf{Q}} = |\mathbf{J}| \cdot \mathbf{Q}, \tag{11}$$

and

$$\begin{bmatrix} \tilde{\mathbf{F}} \\ \tilde{\mathbf{G}} \end{bmatrix} = |\mathbf{J}|\mathbf{J}^{-1} \begin{bmatrix} \mathbf{F} \\ \mathbf{G} \end{bmatrix}. \tag{12}$$

The Jacobian $\mathbf{J}$ is computed for each standard element using

$$\mathbf{J} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix}, \tag{13}$$

where the metrics of the Jacobian are obtained from the relationship between the physical non-uniform element and the computational standard element.

3. A brief review of the SD method

For brevity, we only discuss the SD method for quadrilateral elements. Its numerical formulation and implementation have been reported extensively in [8,7].

In the SD standard element, two sets of points are defined; the solution points and the flux points, which are illustrated in Fig. 1. Note that $\tilde{\mathbf{F}}$ and $\tilde{\mathbf{G}}$ are stored at different sets of flux points. For the 4^{th} order case, 20 flux points are used to store $\tilde{\mathbf{F}}$ and 20 more flux points are used to store $\tilde{\mathbf{G}}$.

In order to construct a degree $(N-1)$ polynomial in each coordinate direction, N solution points are required. The solution points in one dimension are chosen to be the Gauss points,

$$X_s = \frac{1}{2}\left[1 - \cos\left(\frac{2s-1}{2N}\pi\right)\right], \quad s = 1, 2, \ldots, N. \tag{14}$$

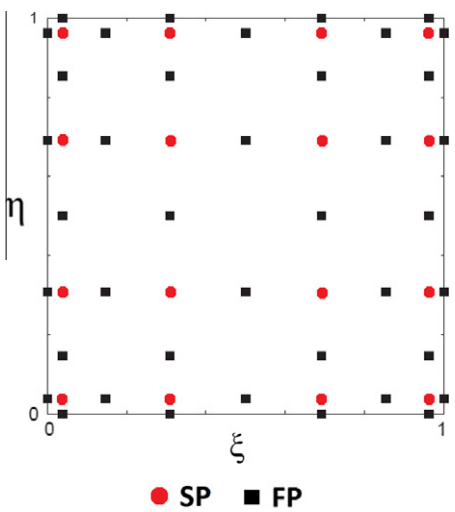

Fig. 1. Distribution of staggered solution points (SP) and flux points (FP) for a 4th order SD method.

We use Legendre–Gauss quadrature points plus two end points 0 and 1 for flux points, as suggested by Huynh [3]. Choosing $P_{-1}(\xi) = 0$ and $P_0(\xi) = 1$, the higher-degree Legendre polynomials can be determined by

$$P_n(\xi) = \frac{2n-1}{n}(2\xi - 1)P_{n-1}(\xi) - \frac{n-1}{n}P_{n-2}(\xi). \tag{15}$$

The locations of these Legendre–Gauss quadrature points are the roots of the equation $P_n(\xi) = 0$. In [8,7], it is shown that the polynomial for flux points dictates the order of accuracy in terms of spatial discretization.

Using the solutions at N solution points, a $(N-1)$ degree polynomial can be built using the Lagrange basis

$$h_i(X) = \prod_{s=1, s\neq i}^{N}\left(\frac{X - X_s}{X_i - X_s}\right). \tag{16}$$

Similarly, using the fluxes at $(N+1)$ flux points, a N degree polynomial can be built using the Lagrange basis

$$l_{i+1/2}(X) = \prod_{s=0, s\neq i}^{N}\left(\frac{X - X_{s+1/2}}{X_{i+1/2} - X_{s+1/2}}\right). \tag{17}$$

The reconstructed solution vector in the 2D standard element is the tensor product of the two one-dimensional polynomials

$$\mathbf{Q}(\xi,\eta) = \sum_{j=1}^{N}\sum_{i=1}^{N}\frac{\tilde{\mathbf{Q}}_{i,j}}{|\mathbf{J}_{i,j}|}h_i(\xi)\cdot h_j(\eta). \tag{18}$$

Similarly, the reconstructed flux vectors are obtained through

$$\tilde{\mathbf{F}}(\xi,\eta) = \sum_{j=1}^{N}\sum_{i=0}^{N}\tilde{\mathbf{F}}_{i+\frac{1}{2},j}\left[l_{i+\frac{1}{2}}(\xi)\cdot h_j(\eta)\right], \tag{19}$$

$$\tilde{\mathbf{G}}(\xi,\eta) = \sum_{j=0}^{N}\sum_{i=1}^{N}\tilde{\mathbf{G}}_{i,j+\frac{1}{2}}\left[h_i(\xi)\cdot l_{j+\frac{1}{2}}(\eta)\right]. \tag{20}$$

The reconstructed fluxes are only element-wise continuous, but discontinuous across cell interfaces. As such, across the interfaces, we employ the simple Rusanov solver suggested by Rusanov [14] to compute the inviscid fluxes and an averaging approach reported in [6] to compute the viscous fluxes.

4. Numerical formulation of the CPR method

Although the number of solution points is the same for both SD and CPR methods, we use a much smaller number of flux points for the latter. Fig. 2 presents a computational element for the 4th order CPR method. Note the collocation of the flux and solution points. Thus, in the CPR method, the vectors $\tilde{\mathbf{F}}$ and $\tilde{\mathbf{G}}$ are stored at an identical set of flux points. This order of

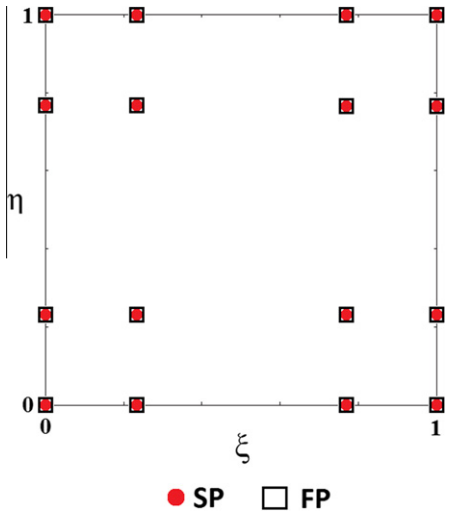

Fig. 2. Distribution of collocated solution points (SP) and flux points (FP) for a 4th order CPR method.

CPR method only requires 16 flux points to store the flux vectors, whereas the 4th order SD method uses a total of 40 flux points per standard element.

According to [4], reconstruction for each cell deals with the jumps at the two interfaces. Thus, a continuous flux function can be defined by making corrections to the discontinuous flux function. Let the polynomial $\tilde{\mathbf{F}}_{j,k}(\xi)$ represent the continuous flux function in ξ direction, where j and k define the cell and solution point, respectively. This function accounts for the data interaction among adjacent cells by taking on common flux values at the two interfaces. Furthermore, let $\tilde{\mathbf{f}}_{j,k}(\xi)$ represent the discontinuous flux function along the same direction where the correction to this function is made by

$$\tilde{\mathbf{F}}_{j,k}(\xi) = \tilde{\mathbf{f}}_{j,k}(\xi) + \left[f^{com}_{j-\frac{1}{2}} - f_j(0)\right] g_{LB}(\xi) + \left[f^{com}_{j+\frac{1}{2}} - f_j(1)\right] g_{RB}(\xi). \tag{21}$$

This equation provides corrections to both left and right boundaries of the standard element, with the correction to the left boundary $g_{LB}(\xi)$, being separate from that at the right boundary $g_{RB}(\xi)$. This formulation can be extended to find an expression for correcting the discontinuous flux function in the η direction, $\tilde{\mathbf{g}}_{j,k}(\eta)$, to compute the continuous flux function $\tilde{\mathbf{G}}_{j,k}(\eta)$. The correction function for the left boundary must take the values

$$g_{LB}(0) = 1, \tag{22}$$

$$g_{LB}(1) = 0, \tag{23}$$

and the correction function for the right boundary must take the values

$$g_{RB}(0) = 0, \tag{24}$$

$$g_{RB}(1) = 1. \tag{25}$$

Both g_{LB} and g_{RB} are of degree N and approximate the zero function. By reflection, $g_{RB}(\xi) = g_{LB}(1-\xi)$ on the standard element interval $I = [0,1]$. The function $\tilde{\mathbf{F}}_j$ is of degree N, takes on the two common flux values

$$\mathbf{F}_j(0) = f^{com}_{j-\frac{1}{2}}, \tag{26}$$

$$\mathbf{F}_j(1) = f^{com}_{j+\frac{1}{2}}, \tag{27}$$

and approximates the discontinuous flux function $\tilde{\mathbf{f}}_j$, which is of degree $N-1$. This correction to the discontinuous flux function ultimately insures continuity across interfaces. Furthermore, a derivative formulation is of the form

$$(\tilde{\mathbf{F}}_{j,k})_\xi(\xi) = (\tilde{\mathbf{f}}_{j,k})_\xi(\xi) + \left[f^{com}_{j-\frac{1}{2}} - f_j(0)\right] g'_{LB}(\xi) + \left[f^{com}_{j+\frac{1}{2}} - f_j(1)\right] g'_{RB}(\xi). \tag{28}$$

This derivative of the continuous flux function is readily obtained once the correction functions, g_{LB} and g_{RB}, and their derivatives are defined over the interval of the standard element. This formulation can be extended to find the derivatives of the continuous flux function in the η direction, $(\tilde{\mathbf{G}}_{j,k})_\eta(\eta)$. Similar to the SD method, a simple Rusanov solver is used to compute the inviscid fluxes across the interface, $f^{com}_{j-\frac{1}{2}}$. However, the BR2 scheme reported in [2] is employed here for computing viscous fluxes in the CPR method. Although this method is different than that used for the SD scheme, the effect on the efficiency of the CPR method is small due to the fact that an explicit time marching method is used.

In summary, the algorithm to compute inviscid flux derivatives in the CPR method consists of the following steps:

(1) Given the conservative variables at the solution points, $\tilde{\mathbf{F}}(\xi)$ and $\tilde{\mathbf{G}}(\eta)$ are directly computed at the flux points. Their derivatives can be computed using a $N-2$ degree polynomial in the corresponding cell to get $\tilde{\mathbf{f}}_\xi$ and $\tilde{\mathbf{g}}_\eta$ respectively.
(2) The inviscid common fluxes (e.g. $f^{com}_{j-\frac{1}{2}}$) at the element interfaces are computed using the Rusanov solver.
(3) The derivatives $\tilde{\mathbf{F}}_\xi$ and $\tilde{\mathbf{G}}_\eta$ can then be computed at the solution points k via flux reconstruction using Eq. (28).

The procedure to get viscous fluxes within the CPR framework is described in the following steps:

(1) Compute the common solution $\mathbf{Q}^{com}_f = \frac{1}{2}(\mathbf{Q}^-_f + \mathbf{Q}^+_f)$ at cell interfaces.
(2) Similar to the procedure of Eq. (28), we can compute the corrected $\nabla\mathbf{Q}_f$ using $\mathbf{R}^- = \nabla\mathbf{Q}^-_f + \mathbf{r}^-_f$ for the left cell and $\mathbf{R}^+ = \nabla\mathbf{Q}^+_f + \mathbf{r}^+_f$ for the right cell.
(3) The common gradient is computed by $\nabla\mathbf{Q}^{com}_f = \frac{1}{2}\left(\nabla\mathbf{Q}^-_f + \mathbf{r}^-_f + \nabla\mathbf{Q}^+_f + \mathbf{r}^+_f\right)$.
(4) The viscous flux derivatives $\tilde{\mathbf{F}}^v_\xi$ and $\tilde{\mathbf{G}}^v_\eta$ can then be computed at the solution points k via flux reconstruction using

$$\left(\tilde{\mathbf{F}}^v_{j,k}\right)_\xi(\xi) = \tilde{\mathbf{f}}^v_\xi(\mathbf{Q},\mathbf{R}) + \left[f^v_{j-\frac{1}{2},com} - f^v_j(0)\right] g'_{LB}(\xi_k) + \left[f^v_{j+\frac{1}{2},com} - f^v_j(1)\right] g'_{RB}(\xi_k). \tag{29}$$

The above procedure of computing viscous fluxes is an analogy to the BR2 scheme proposed by Bassi et al. [2]. This discretization method, the I-continuous method proposed by Huynh [4], and the LDG2 method proposed by Kannan and Wang [5] are compact in the sense that they involve information from direct neighboring cells only.

The current choice of correction function is $g = g_2$, which is defined in [4] and is also known as $g_{Lump,Lo}$. This choice is motivated by the fact that the derivative of this function vanishes at all solution points except at the left boundary if evaluating g'_{LB} or at the right boundary if evaluating g'_{RB}. As it turns out, the zeros of g'_2 are the Legendre–Lobatto points. Therefore, it is very convenient to choose the Lobatto points as the solution points. With this choice, the jump in flux at the left interface results in a correction to only $\tilde{f}_\xi$ evaluated at the interface. The correction to $\tilde{f}_\xi$ at all other solution points is zero. The g_2 function is defined as

$$g_2 = \frac{N-1}{2N-1}R_{R,N} + \frac{N}{2N-1}R_{R,N-1}, \tag{30}$$

where $R_{R,N}$ represents the right Radau polynomial

$$R_{R,N} = \frac{(-1)^N}{2}(P_N - P_{N-1}), \tag{31}$$

and P_N is the Legendre polynomial. On the interval $I = [0,1]$, for the left boundary

$$g'_{LB}(0) = g'_2(0) = N(1-N), \tag{32}$$

$$g'_{LB}(1) = g'_2(1) = 0. \tag{33}$$

For the right boundary

$$g'_{RB}(0) = g'_2(0) = 0, \tag{34}$$

$$g'_{RB}(1) = g'_2(1) = N(N-1). \tag{35}$$

4.1. Time marching scheme

All computations utilize a 4th order accurate, strong-stability-preserving five-stage Runge–Kutta time marching scheme introduced by Spiteri and Ruuth [15].

5. Euler vortex case

To demonstrate the order of accuracy for both the SD and CPR method, we perform a side-by-side study of a moving inviscid compressible vortex for which there exists an analytical solution. The vortex is initially centerd at $(x_o, y_o) = (0,0)$ within a domain sized 20×15 and set to move at an angle $tan^{-1}(\frac{1}{2})$, with respect to the free stream, to a position located at $(x,y) = (5.0, 2.5)$. Periodic boundary conditions have been applied for this case. The strength and diameter of the vortex are 4.0 and 2.0, respectively. The freestream Mach number is 0.5. We compute the L_2-norm error over various mesh sizes for both 3^{rd} and 4^{th} methods and report them in Tables 1 and 2. It should be noted for this study that the CFL $(\frac{max(\Delta t)U_\infty N^2}{\Delta x})$ limits for third-order SD and CPR methods are approximately 1.152 and 1.314 respectively, where U_∞ is the freestream velocity.

6. Validation of numerical solvers

Fig. 3 presents a mesh with all quadrilateral elements for a NACA0012 airfoil. The grid has 6345 cells in all and 94 edges to represent the profile of this airfoil. The curvature of its profile is improved using a cubic spline fitting method.

The freestream Mach number is fixed at 0.2 for all computations. The Reynolds number is set to 500 for viscous flow cases. No-slip and adiabatic wall boundary condition is employed for all computations of viscous flow. The drag coefficients are 0.173 predicted by both the CPR and SD solver using either 4th order or 5th order methods. It demonstrates that the grid

Table 1
SD method L2 error and order of accuracy for Euler vortex flow using 3rd & 4th order schemes.

N order	No. of Elements	L2 error	CPR order
3rd	300	1.121E−3	–
	1200	1.703E−4	2.72
	4800	2.363E−5	2.85
	19200	3.677E−6	2.68
4th	300	3.324E−4	–
	1200	2.278E−5	3.87
	4800	9.444E−6	4.59
	19200	8.777E−8	3.43

Table 2
CPR method L2 error and order of accuracy for Euler vortex flow using 3rd & 4th order schemes.

N order	No. of Elements	L2 error	SD Order
3rd	300	2.466E−3	–
	1200	4.700E−4	2.40
	4800	7.084E−5	2.73
	19200	1.104E−5	2.68
4th	300	8.178E−4	–
	1200	6.620E−5	3.63
	4800	4.597E−6	3.85
	19200	3.239E−7	3.83

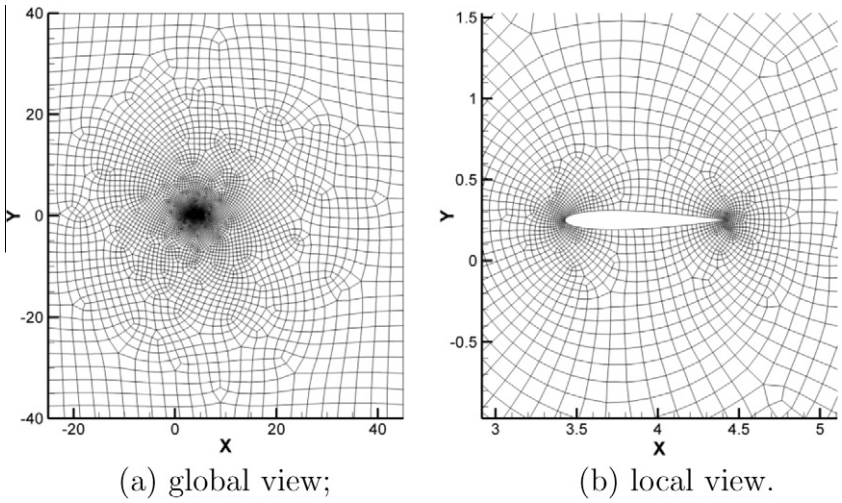

Fig. 3. A quadrilateral mesh for NACA0012 airfoil.

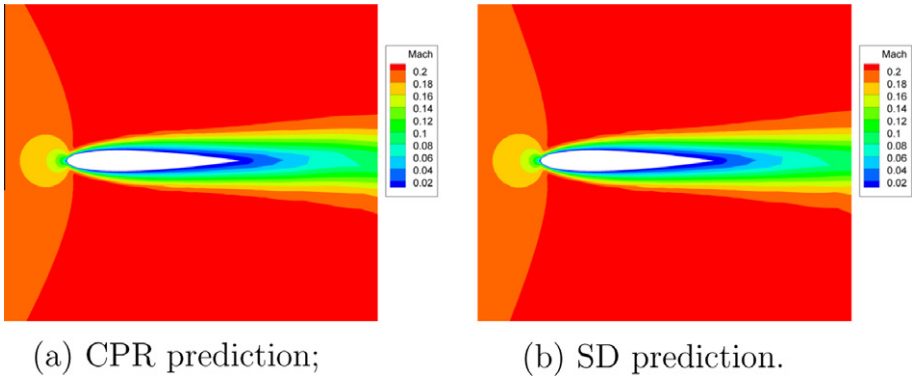

Fig. 4. Mach number contours predicted by viscous flow solvers.

resolution is sufficient even for the 4th order simulations. Note that [11] predicted $C_d = 0.172$ using the PowerFlow software. Fig. 4 presents the Mach number contours predicted by the CPR and SD solvers respectively for the viscous flow at Reynolds number 500. Consistent with the quantitative results from the Euler vortex case, the results here from SD and CPR agree quite well.

In addition, for inviscid flow cases the CPR and SD methods both predicted drag coefficients smaller than 10^{-3}. Although the source code is not ready to be distributed as open source software at this point in time, we would be willing to provide the source code to those who have a research need.

Table 3
CPU time of CPR and SD solvers for inviscid flow past a NACA0012 airfoil (in seconds).

Order	Runs	R–K iterations	CPR (s)	SD (s)
4th	No. 1	900	256.4	355.3
	No. 2	900	256.6	357.8
5th	No. 1	900	167.2	223.9
	No. 2	900	170.3	221.7

Table 4
CPU time of CPR and SD solvers for viscous flow past a NACA0012 airfoil (in seconds).

Order	Runs	R–K iterations	CPR (s)	SD (s)
4th	No. 1	1000	463.5	650.1
	No. 2	1000	459.7	653.8
	No. 3	1000	453.2	649.1
5th	No. 1	1000	703.1	1015.8
	No. 2	1000	696.9	1049.3
	No. 3	1000	706.6	1031.3

7. Comparison of processing time

As stated previously, the CPR method employs fewer flux points. As a result, CPR reduces the number of operations in computing $\tilde{\mathbf{F}}(\mathbf{Q},\mathbf{R})$ and $\tilde{\mathbf{G}}(\mathbf{Q},\mathbf{R})$. The 4th order CPR method computes $\tilde{\mathbf{F}}(\mathbf{Q},\mathbf{R})$ on 16 flux points in ξ direction and $\tilde{\mathbf{G}}(\mathbf{Q},\mathbf{R})$ on 16 flux points in η direction. The 4^{th} order SD method employs 20 flux points in each direction. Therefore, the SD method is 25% more expensive in computing $\tilde{\mathbf{F}}(\mathbf{Q},\mathbf{R})$ and $\tilde{\mathbf{G}}(\mathbf{Q},\mathbf{R})$. In addition, the g_2 method avoids extrapolating conservative variables $\tilde{\mathbf{Q}}$ of solution points onto flux points. Both CPR and SD methods require a similar operation count to compute common inviscid/viscous fluxes at cell interfaces where they have the same number of flux points. The primary cost in computing derivatives using the CPR method is dictated by a degree $(N-2)$ polynomial added with a small cost relevant to the correction (e.g. Eq. 28). Therefore, CPR is more economical than SD, which depends on a degree $N-1$ polynomial for computing derivatives.

One of the primary goals of this paper is to conduct a comparison between two modern discontinuous high-order methods based on differential form. Both solvers use identical time marching schemes. Both solvers are programmed using very similar data structures and identical programming languages and compilers. The simulations are run on a DELL T7500 workstation based serial processing. We have used the same time step size for both CPR and SD solvers for every test case. The goal is to study the cost of CPR and SD on the evaluation of residual. However, evaluating nonlinear flux terms could be as costly as, if not more expensive than, the interpolation/extrapolation of Q and F between flux and solution points in the SD method. [3], and more recently [17], showed that CFL limits vary across a range of CPR schemes as well as the standard SD scheme.

Table 3 compares two inviscid flow solvers using the CPR and SD methods. For the 4th order scheme, the CPR approach is approximately 27% faster than the SD method. For the 5th order scheme, CPR is about 35% faster than SD. These results show an improvement in efficiency for a successively higher-order spatial discretization of CPR. The computational efficiency of the CPR method is more promising for viscous flows. Table 4 compares the CPU time of both CPR and SD methods in solving viscous flow past an airfoil. When the fourth-order methods are used, CPR is approximately 42% faster than SD. Moreover, the fifth-order CPR is about 45% faster than the SD method.

8. Concluding remark

We successfully developed a 2D solver using the CPR method on unstructured quadrilateral meshes for compressible Navier–Stokes equations. We compared the computational efficiency of the CPR method to that of the SD method. The order of accuracy of both methods has been demonstrated by a study of a moving inviscid vortex. By simulating steady flow over a NACA0012 airfoil, we show that the 4th order CPR method is approximately 27% faster than the SD method for inviscid flow. For viscous flow, the 4th order CPR method is nearly 42% faster than the SD method. Furthermore, results demonstrate the advantage of the CPR method over the SD method in terms of computational efficiency for higher-order accurate spatial discretizations.

Acknowledgement

Chunlei Liang would like to acknowledge a research grant from ONR with award No. N000141210500.

References

[1] A. Balan, G. May, J. Schoberl, A stable high-order spectral difference method for hyperbolic conservation laws on triangular elements, J. Comput. Phys. 231 (2012) 2359–2375.
[2] F. Bassi, A. Crivellini, S. Rebay, M. Savini, Discontinuous galerkin solution of the reynolds-averaged Navier–Stokes and $k-\omega$ turbulence model equations, Comput. Fluids 34 (2005) 507–540.
[3] H. Huynh, A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods, AIAA Paper AIAA-2007-4079, 2007.
[4] H. Huynh, A reconstruction approach to high-order schemes including discontinuous Galerkin for diffusion, AIAA Paper AIAA-2009-403, 2009.
[5] R. Kannan, Z.J. Wang, LDG2: a variant of the LDG viscous flux formulation for the spectral volume method, J. Sci. Comput. 46 (2011) 314–328.
[6] D.A. Kopriva, A staggered-grid multidomain spectral method for the compressible Navier–Stokes equations, J. Comput. Phys. 143 (1998) 125–158.
[7] C. Liang, A. Jameson, Z.J. Wang, Spectral difference method for two-dimensional compressible flow on unstructured grids with mixed elements, J. Comput. Phys. 228 (2009) 2847–2858.
[8] C. Liang, S. Premasuthan, A. Jameson, High-order accurate simulation of low-mach laminar flow past two side-by-side cylinders using spectral difference method, Comput. Struct. 87 (2009) 812–817.
[9] C. Liang, S. Premasuthan, A. Jameson, Z.J. Wang, Large eddy simulation of compressible turbulent channel flow with spectral difference method, AIAA Paper AIAA-2009-402, 2009.
[10] Y. Liu, M. Vinokur, Z.J. Wang, Spectral difference method for unstructured grids I: basic formulation, J. Comput. Phys. 216 (2006) 780–801.
[11] D.P. Lockard, L.S. Luo, S.D. Milder, B.A. Singer, Evaluation of PowerFLOW for aerodynamic applications, J. Stat. Phys. 107 (2002) 423–478.
[12] G. May, On the connection between the spectral difference method and the discontinuous galerkin method, Commun. Comput. Phys. 9 (2011) 1071–1080.
[13] A.H. Mohammad, Z.J. Wang, C. Liang, LES of turbulent flow past a cylinder using spectral difference method, Adv. Appl. Math. Mech. 2 (2010) 451–466.
[14] V.V. Rusanov, Calculation of interaction of non-steady shock waves with obstacles, J. Comput. Math. Phys. USSR 1 (1961) 267–279.
[15] R.J. Spiteri, S.J. Ruuth, A new class of optimal high-order strong-stability-preserving time discretization methods, SIAM J. Numer. Anal. 40 (2002) 469–491.
[16] Y. Sun, Z.J. Wang, Y. Liu, High-order multidomain spectral difference method for the Navier–Stokes equations on unstructured hexahedral grids, Commun. Comput. Phys. 2 (2007) 310–333.
[17] P.E. Vincent, P. Castonguay, A. Jameson, Insights from von Neumann analysis of high-order flux reconstruction schemes, J. Comput. Phys. 230 (2011) 8134–8154.
[18] P.E. Vincent, P. Castonguay, A. Jameson, A new class of high-order energy stable flux reconstruction schemes, J. Sci. Comput. 47 (2011) 50–72.
[19] Z.J. Wang, H. Gao, A unifying lifting collocation penalty formulation including the discontinuous galerkin, spectral volume/difference methods for conservation laws on mixed grids, J. Comput. Phys. 228 (2009) 8161–8186.
[20] Z.J. Wang, H. Gao, T. Haga, A Unifying Discontinuous Formulation for Hybrid Meshes, World Scientific Publishing, 2011.
[21] Z.J. Wang, Y. Liu, G. May, A. Jameson, Spectral difference method for unstructured grids II: extension to the Euler equations, J. Comput. Phys. 32 (2007) 45–71.