A Parallel Performance Study of the High-order Compact Direct Flux Reconstruction Method for Conservation Laws on Maya Cluster

Lai Wang[1], Meilin Yu[1], and Matthias K. Gobbert[2]

[1]Department of Mechanical Engineering, University of Maryland, Baltimore County

[2]Department of Mathematics and Statistics, University of Maryland, Baltimore County

Technical Report HPCF–2017–1, hpcf.umbc.edu > Publications

Abstract

The compact direct flux reconstruction method (CDFR) for conservation laws utilizes techniques from compact finite difference methods to directly approximate spatial derivatives of fluxes within standard elements. The CDFR scheme is a compact high-order method family which can be efficiently parallelized for high performance computing. In the present study, a parallel performance study of the 3rd-order CDFR scheme with a 3rd-order explicit Runge-Kutta scheme is conducted. The inviscid isentropic vortex propagation problem is adopted as a test case. The numerical performance studies have demonstrated that the CDFR method can efficiently solve conservation laws. The parallel performance study shows excellent observed speedup and efficiency. A comparison between different partition approaches of the mesh also demonstrates that optimized communication between processes can improve the parallel performance.

1 Introduction

Computational fluid dynamics (CFD) can be time-consuming especially when it comes to turbulence simulations. The diversities of both temporal and spatial scales result in small mesh size and time step size to capture the turbulent structures. In terms of both hardware and time cost, it is inevitable to make use of high performance computing in CFD. In terms of numerical methods, we also want develop high-accuracy methods [1] which can be efficiently parallelized for high performance computing.

First proposed by Huynh [2, 3], the flux reconstruction/correction procedure via reconstruction (FR/CPR) method has demonstrated its capacity to handle complex flow problems on unstructured meshes [4]. Correction functions are essential for original FR/CPR schemes. With k solution points, if only Lagrange interpolation is used, a flux polynomial of degree $(k-1)$ can be obtained. Hence, the spatial flux derivatives can only reach $(k-1)$st-order of accuracy. By correcting the local fluxes with correction functions of degree k (or equivalently, a kth-order correction of flux derivatives), the kth-order of accuracy can be recovered. Recently, Romero et al. [5, 6] proposed a simplified FR/CPR, called direct flux reconstruction (DFR), which can obtain kth-order of accuracy schemes without correction functions. Instead, direct polynomial fitting involving all k solution points and 2 flux points within an element is used. The compact direct flux reconstruction method (CDFR) [7] utilize the techniques of compact finite difference schemes [8] within an elements and directly approximate the spatial derivatives of fluxes within an elements. Compared to DFR, the reconstruction of flux polynomials is implicit, i.e., the spatial derivatives are obtained under the premise that there exist corresponding differentiable flux polynomials. It is observed that the flexibility to recover different methods of the CDFR method is hindered compared to the original FR/CPR method, due to the fact that only Gaussian points within the standard element can assure CDFR schemes stable and no correction functions are used. However, the CDFR method needs less calculation to obtain the spatial derivatives than the FR/CPR method. This fact potentially makes CDFR to be more efficient.

In the present study, the two dimensional (2D) Euler equation is solved. The CDFR methods are employed for the spatial discretization and the 3rd-order three stage Runge-Kutta scheme is used for temporal discretization. The parallel performance study is conducted on the HPCF2013 portion of the maya cluster, the UMBC High Performance Computing Facility (HPCF). HPCF2013 has 72 nodes. Every node has two Intel E5-2650v2 Ivy Bridge (2.6 GHz, 20 MB cache), processors with eight cores apiece, i.e., 16 cores per node. For HPCF2013, a high performance quad data rate (DQR) InfiniBand (IB) interconnect, termed as IB-QDR, connects all nodes. Ideally the system can achieve a latency of 1.2 μs to transfer a message between two nodes, and can support a bandwidth of up to 40 Gbps (Gigabits per second). All nodes are running Red Hat Enterprise Linux 6.4. g++ (GCC) 4.8.4 with Intel(R) MPI Library for Linux* OS, 64-bit applications, Version 5.0 Update 3 Build 20150128 is used as the compiler. The `-std=c++11` compiler option is chosen to make use of the C++11 standard of C++ programming language.

The remainder of this report is organized as follows. Section 2 gives a brief introduction of the CDFR method. Then Section 3 elaborates the 2D Euler equation and explains how to extend CDFR to solve the 2D Euler equation. In Section 4, we discuss the parallel implementations. Section 5 demonstrates the numerical results and the parallel performances study. The last section draws conclusions for the present study.

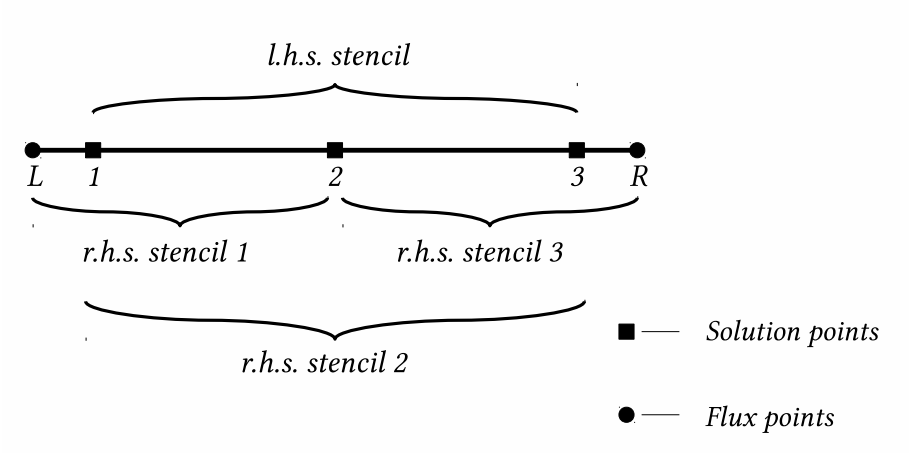


Figure 2.1: Illustration of the l.h.s. and three r.h.s. stencils used in the 3rd-order CDFR scheme.

2 A Brief Introduction of the Compact Direct Flux Reconstruction Method

Consider the one-dimensional (1D) conservation law,

$$\frac{\partial u}{\partial t}+\frac{\partial f}{\partial x}=0, \tag{2.1}$$

defined on the space-time domain $\Omega\times[0,+\infty)$.

When the CDFR method is employed to solve (2.1), the spatial domain Ω is divided into N non-overlapping intervals $\Omega_j, j=1,2,...,N$. The reconstruction of the CDFR method is conducted within the standard element. In the present study, the physical element $\Omega_j, x\in[x_{j-1/2},x_{j+1/2}]$ is projected onto a standard element $\xi\in[-1,1]$. For this coordinate transformation,

$$f_x=f_\xi\xi_x, \xi_x=\Delta\xi/\Delta x, \Delta\xi=2. \tag{2.2}$$

Since the reconstruction of derivatives is conducted with respect to ξ, the subscript of coordinate in the spatial derivatives is neglected in the followings. The kth-order CDFR method makes use of k Gauss points(solution points) $\xi_i, i=1,2,\ldots,k$ and two ending points(flux points), $\xi_{L/R}=-1/1$, within the standard element. L and R are the left ending point and right ending point respectively. The utilization of the ending points assures the conservation of the method. For simplicity of notation, we use $i=1,2,\ldots,k$ as the index of spatial derivative on the solution points, such as f_i' and $i=L,1,2,\ldots,k,R$ for flux on all the points, such as f_i. A kth-order finite difference scheme within the standard element (in terms of spatial derivatives of Gauss points $i, i=1,2,\ldots,k$) can be expressed as

$$\sum_{l=1}^{l=k}\alpha_l f_l'=\frac{1}{\Delta\xi}\sum_{r=i-1}^{r=i+1}\beta_r f_r+O(\xi_i^k). \tag{2.3}$$

Note that in the left hand side (l.h.s.) of (2.3) all the Gauss points are involved and in the right hand side (r.h.s.) only three consecutive points are involved. And for the right hand hand side, there will be k optional stencils, such as $r=\{L,1,2\},\{1,2,3\},\ldots,\{k-1,k,R\}$. As illustrated in Figure 2.1, in each element Ω_j, taking three solution points $\xi_i, i=1,2,3$, as an example, the l.h.s. of (2.3) will always have a three-point stencil $f(\xi_i), i=1,2,3$. For the r.h.s., there will be three stencils. Specifically, these three discretization formulas can be expressed as follows:

$$\begin{cases}\alpha_{11}f_1'+\alpha_{12}f_2'+\alpha_{13}f_3'=\frac{1}{\Delta\xi}(\beta_{1L}f_{com,L}+\beta_{11}f_1+\beta_{12}f_2),\\ \alpha_{21}f_1'+\alpha_{22}f_2'+\alpha_{23}f_3'=\frac{1}{\Delta\xi}(\beta_{21}f_1+\beta_{22}f_2+\beta_{23}f_3),\\ \alpha_{31}f_1'+\alpha_{32}f_2'+\alpha_{33}f_3'=\frac{1}{\Delta\xi}(\beta_{32}f_2+\beta_{33}f_3+\beta_{3R}f_{com,R}),\end{cases} \tag{2.4}$$

where $f_{com,L}$ and $f_{com,R}$ are common fluxes at left(L) and right(R) interfaces of Ω_j, respectively. For $\alpha_{li},\beta_{lj},i,l=1,2,3,j=L,1,2,3,R$, as mentioned above, Taylor expansion can be used to obtain their values. To be more specific, taking stencil $l=1$ as an example, the coefficients of $f^{(n)}$s on each side should be consistent up to $n=4$. Thus, we

can formulate a linear system for the first stencil as

$$\begin{pmatrix} 0 & 0 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & -\frac{1}{2}\xi_L & -\frac{1}{2}\xi_1 & -\frac{1}{2}\xi_2 \\ \xi_1 & \xi_2 & \xi_3 & -\frac{1}{2}\frac{\xi_L{}^2}{2!} & -\frac{1}{2}\frac{\xi_1{}^2}{2!} & -\frac{1}{2}\frac{\xi_2{}^2}{2!} \\ \frac{\xi_1{}^2}{2!} & \frac{\xi_2}{2!} & \frac{\xi_3}{2!} & -\frac{1}{2}\frac{\xi_L{}^3}{3!} & -\frac{1}{2}\frac{\xi_1{}^3}{3!} & -\frac{1}{2}\frac{\xi_2{}^3}{3!} \\ \frac{\xi_1{}^3}{3!} & \frac{\xi_2{}^3}{3!} & \frac{\xi_3{}^3}{3!} & -\frac{1}{2}\frac{\xi_L{}^4}{4!} & -\frac{1}{2}\frac{\xi_1{}^4}{4!} & -\frac{1}{2}\frac{\xi_2{}^4}{4!} \\ 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha_{11} \\ \alpha_{12} \\ \alpha_{13} \\ \beta_{1L} \\ \beta_{11} \\ \beta_{12} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}, \tag{2.5}$$

where the 6th equation is the restriction $\sum_{i=1}^{i=3} \alpha_{1i} = 1$ to avoid trivial solutions. Note that $\xi_L = -1$ and $\xi_R = 1$. Here we use

$$\boldsymbol{M}_1 \boldsymbol{C}_1 = \boldsymbol{B} \tag{2.6}$$

to represent the linear system (2.5). Similar forms for the second and third stencils in (2.4) can be formulated. As a result, the linear systems used to obtain the coefficients in (2.4) can be expressed as follows:

$$\begin{cases} \boldsymbol{M}_1 \boldsymbol{C}_1 = \boldsymbol{B}, \\ \boldsymbol{M}_2 \boldsymbol{C}_2 = \boldsymbol{B}, \quad . \\ \boldsymbol{M}_3 \boldsymbol{C}_3 = \boldsymbol{B}. \end{cases} \tag{2.7}$$

By solving (2.7), we can obtain all the coefficients $\alpha_{li}, \beta_{lj}, i, l = 1, 2, 3, j = L, 1, 2, 3, R$. Finally, (2.4) can be rewritten as

$$\boldsymbol{f'} = \frac{1}{\Delta\xi} \boldsymbol{\alpha}^{-1} \boldsymbol{\beta} \tilde{\boldsymbol{f}}. \tag{2.8}$$

where $\boldsymbol{\alpha}$ is a 3×3 matrix and $\boldsymbol{\beta}$ is a 3×5 matrix,

$$\boldsymbol{\alpha} = \begin{pmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} \\ \alpha_{31} & \alpha_{32} & \alpha_{33} \end{pmatrix}, \quad \boldsymbol{f'} = \begin{pmatrix} f_1' \\ f_2' \\ f_3' \end{pmatrix}, \tag{2.9}$$

$$\boldsymbol{\beta} = \begin{pmatrix} \beta_{1L} & \beta_{11} & \beta_{12} & 0 & 0 \\ 0 & \beta_{21} & \beta_{22} & \beta_{23} & 0 \\ 0 & 0 & \beta_{32} & \beta_{33} & \beta_{3R} \end{pmatrix}, \quad \tilde{\boldsymbol{f}} = \begin{pmatrix} f_{com,L} \\ f_1 \\ f_2 \\ f_3 \\ f_{com,R} \end{pmatrix}. \tag{2.10}$$

A common flux solver which satisfies physical conservation laws must be employed to reconstruct the common fluxes on interfaces. For (2.1), an upwind Riemann solver is preferred. For example, the Roe Riemann solver or Rusanov Riemann solver can be adopted for Euler equations. Particularly, for the 1D first-order wave equation (i.e., $f = cu$ with c a constant in (2.1)), the left common flux of the current element Ω_j can be expressed as

$$f_{com,L}^j = \begin{cases} f_R^{j-1}, & if\ c > 0, \\ f_L^j, & if\ c < 0. \end{cases} \tag{2.11}$$

Since the common flux solver involves neighboring elements, we introduce superscript 'j' to denote the jth element. Lagrange interpolation is used to obtain local fluxes f_L^j and f_R^j on interfaces as follows

$$f_{L/R}^j = \sum_{i=1}^{3} \left(\left(\prod_{l=1, l\neq i}^{3} \frac{\xi_{L/R} - \xi_l}{\xi_i - \xi_l} \right) f_i^j \right), \quad \xi_{L/R} = -1/1. \tag{2.12}$$

To construct the arbitrary order CDFR method, the procedure is similar. See details in Ref. [7]. The analysis of accuracy, stability, dissipation and dispersion properties of the CDFR method can also be found in Ref. [7].

3 Extension to the 2D Euler Equation

The 2D Euler equations can be expressed as

$$\frac{\partial \boldsymbol{q}}{\partial t} + \frac{\partial \boldsymbol{f}}{\partial x} + \frac{\partial \boldsymbol{g}}{\partial y} = 0, \tag{3.1}$$

where

$$\boldsymbol{q} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix}, \quad \boldsymbol{f} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (E+p)u \end{pmatrix}, \quad \boldsymbol{g} = \begin{pmatrix} \rho u \\ \rho uv \\ \rho v^2 + p \\ (E+p)v \end{pmatrix}, \tag{3.2}$$

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

$$E = \frac{p}{\gamma - 1} + \frac{1}{2}\rho(u^2 + v^2), \tag{3.3}$$

with the ratio of specific heat $\gamma = 1.4$. After transforming (3.1) into the computational domain (i.e., the standard element $[-1, 1] \times [-1, 1]$) (ξ, η), we have

$$\frac{\partial \boldsymbol{Q}}{\partial t} + \frac{\partial \boldsymbol{F}}{\partial \xi} + \frac{\partial \boldsymbol{G}}{\partial \eta} = 0, \tag{3.4}$$

where

$$\begin{cases} \boldsymbol{Q} = |\boldsymbol{J}|\boldsymbol{q}, \\ \boldsymbol{F} = |\boldsymbol{J}|(\boldsymbol{f}\xi_x + \boldsymbol{g}\xi_y), \\ \boldsymbol{G} = |\boldsymbol{J}|(\boldsymbol{f}\eta_x + \boldsymbol{g}\eta_y), \end{cases} \tag{3.5}$$

and

$$\boldsymbol{J} = \frac{\partial(x, y)}{\partial(\xi, \eta)}, \quad |\boldsymbol{J}| = x_\xi y_\eta - x_\eta y_\xi. \tag{3.6}$$

If no moving grids are involved, (3.4) can be rewritten as follows

$$\frac{\partial \boldsymbol{q}}{\partial t} + \frac{1}{|\boldsymbol{J}|}\left(\frac{\partial \boldsymbol{F}}{\partial \xi} + \frac{\partial \boldsymbol{G}}{\partial \eta}\right) = 0. \tag{3.7}$$

On interfaces of quadrilateral elements, $\boldsymbol{F}$ and $\boldsymbol{G}$ in (3.5) can be rewritten as

$$\begin{cases} \boldsymbol{F} = |\boldsymbol{J}||\nabla\xi|\boldsymbol{f}_{\boldsymbol{n}}\,\mathrm{sign}(\boldsymbol{n} \cdot \nabla\xi), \\ \boldsymbol{G} = |\boldsymbol{J}||\nabla\eta|\boldsymbol{f}_{\boldsymbol{n}}\,\mathrm{sign}(\boldsymbol{n} \cdot \nabla\eta), \end{cases} \tag{3.8}$$

where

$$|\nabla\xi| = \frac{1}{|\boldsymbol{J}|}\sqrt{x_\eta^2 + y_\eta^2}, \quad |\nabla\eta| = \frac{1}{|\boldsymbol{J}|}\sqrt{x_\xi^2 + y_\xi^2}, \tag{3.9}$$

and $\boldsymbol{n} = (n_x, n_y)$ is the unit normal vector at the interface. The sign function is defined as

$$\mathrm{sign}\,(a) = \begin{cases} 1, & if\ a > 0, \\ -1, & if\ a < 0. \end{cases} \tag{3.10}$$

Herein, $\boldsymbol{f}_{\boldsymbol{n}}$ is the normal flux in the physical domain.

$$\boldsymbol{f}_{\boldsymbol{n}} = \boldsymbol{f}n_x + \boldsymbol{g}n_y. \tag{3.11}$$

Note that common fluxes are always calculated in the physical domain, As a result, $\boldsymbol{F}_{com}$ and $\boldsymbol{G}_{com}$ can be expresses as

$$\begin{cases} \boldsymbol{F}_{com} = |\boldsymbol{J}||\nabla\xi|\boldsymbol{f}_{com,\boldsymbol{n}}\,\mathrm{sign}(\boldsymbol{n} \cdot \nabla\xi), \\ \boldsymbol{G}_{com} = |\boldsymbol{J}||\nabla\eta|\boldsymbol{f}_{com,\boldsymbol{n}}\,\mathrm{sign}(\boldsymbol{n} \cdot \nabla\eta), \end{cases} \tag{3.12}$$

where $\boldsymbol{f}_{com,\boldsymbol{n}}$ is the common flux in $\boldsymbol{n}$ direction. $\boldsymbol{f}_{com,\boldsymbol{n}}$ is given as

$$\boldsymbol{f}_{com,\boldsymbol{n}} = \frac{1}{2}\left(\boldsymbol{f}_{\boldsymbol{n},L} + \boldsymbol{f}_{\boldsymbol{n},R} - \boldsymbol{R}|\boldsymbol{\Lambda}|\boldsymbol{R}^{-1}(\boldsymbol{q}_R - \boldsymbol{q}_L)\right). \tag{3.13}$$

Given that the Jacobian matrix at the interface is $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{q}}n_x + \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{q}}n_y$, $|\boldsymbol{\Lambda}|$ is the diagonal matrix with absolute values of averaged eigenvalues and $\boldsymbol{R}$ is a matrix whose columns are the corresponding averaged eigenvectors. Herein, the Roe average is used to calculate $\boldsymbol{R}$, $\boldsymbol{R^{-1}}$, and $|\boldsymbol{\Lambda}|$. The subscripts 'L' and 'R' denote the left side and right side of the current interface, respectively. Note that the 'L' side of the interface is always the current element.

After the transformation, the CDFR method can be used in a dimension by dimension sense for the spatial derivatives in (3.7). For temporal discretization of (3.7), the third-order three-stage strong stability-preserving Runge-Kutta scheme (SSPRK3) [9] is used, which can be written as

$$\begin{cases} \boldsymbol{q}^{(1)} = \boldsymbol{q}^n + \Delta t L(\boldsymbol{q}^n), \\ \boldsymbol{q}^{(2)} = \frac{3}{4}\boldsymbol{q}^n + \frac{1}{4}\boldsymbol{q}^{(1)} + \frac{1}{4}\Delta t L(\boldsymbol{q}^{(1)}), \\ \boldsymbol{q}^{n+1} = \frac{1}{3}\boldsymbol{q}^n + \frac{2}{3}\boldsymbol{q}^{(2)} + \frac{2}{3}\Delta t L(\boldsymbol{q}^{(2)}), \end{cases} \tag{3.14}$$

where $L(\boldsymbol{q}) = -\frac{1}{|\boldsymbol{J}|}\left(\frac{\partial \boldsymbol{F}}{\partial \xi} + \frac{\partial \boldsymbol{G}}{\partial \eta}\right)$, subscript '$n$' denotes the current time step and '$n+1$' denotes the next time step.

4 Parallel Implementation for Unstructured Grids

Unlike structured grids, unstructured girds require the connectivity of different elements in the entire domain. In most cases, the neighboring elements of element m will not be $m-1$ or $m+1$. The partitioning of unstructured grids for parallel computing needs special intention. A number of softwares have been developed for the partition of unstructured grids [10]. In the present study, the domain of the test case is a square and the CDFR method only works on quadrilateral elements. The entire domain is uniformly divided into $N \times N$ non-overlapping elements. Therefore, we can manually and uniformly divide the domain based on the number of processes is used. For complicated geometries and meshes, we can seek help from well-developed softwares [10]. Different partition approaches are tested in this report, i.e., dividing the mesh in only x direction and dividing the mesh in both x and y directions (always divide the mesh in x direction into more parts when the number of processes is not the square of any integer). As illustrated in Figure 4.1, if dividing the mesh in both x and y direction, taking eight processes as an example, on each process, connectivity information of the local unstructured mesh is stored as well as the connectivity information with respect to elements in neighboring processes. Simply from a geometric sense, when the mesh is divided in both x and y directions, the total communication cost will be lower that only dividing the mesh in x direction with the same number of processes. In the procedure of the CDFR methods, when calculating the common fluxes, information from neighboring cells are required. To be more specific, the elements that contains the surfaces shared by different processes need the information from other processes. Thus, we can treat these shared faces as ones that have special boundary conditions regarding the parallel mesh-partition. As for the communication between different processes, we wrap up the information on all the shared faces and use `MPI_Send()` and `MPI_Recv()` to send and receive them together. Since the feature of unstructured grids also exists in these shared faces, we need use blocking send and receive MPI commands which can make sure all the shared-boundary information are received before calculating the spatial derivatives in the local mesh on each process. Besides, `MPI_Allreduce()` is used for the calculation of global error and global residual.

5 Numerical Results and the Performance Study

In this session, an irentropic vortex propagation case is used to study the numerical performance and parallel performance of the CDFR scheme. The isentropic vortex propagation describes the superposition of an isentropic vortex of strenth Γ and an uniform flow of velocities u_∞ and v_∞. The exact solution of the vortex propagation when $t < 9$ can be given as [5]

$$\begin{cases} \rho_{exact} = (1 - \frac{\Gamma^2(\gamma-1)}{8\gamma\pi^2}e^{2f})^{\frac{1}{\gamma-1}}, \\ u_{exact} = u_\infty - \frac{\Gamma(y-y_0)}{2\pi}e^f, \\ v_{exact} = v_\infty + \frac{\Gamma(x-x_0)}{2\pi}e^f, \\ p_{exact} = (1 - \frac{\Gamma^2(\gamma-1)}{8\gamma\pi^2}e^{2f})^{\frac{\gamma}{\gamma-1}}, \end{cases} \tag{5.1}$$

where

$$\begin{cases} f = \frac{1}{2}\left(1 - (x-x_c)^2 - (y-y_c)^2\right), \\ x_c = x_0 + u_\infty t, \\ y_c = y_0 + v_\infty t. \end{cases} \tag{5.2}$$

Herein, (x_c, y_c) is the position of the vortex center. For this case, we set $\Gamma = 5, u_\infty = v_\infty = 1$. At $t = 0$, the vortex center is at $(x_0, y_0) = (0, 0)$. The domain Ω is $[-10, 10] \times [-10, 10]$. Periodic boundary conditions are used for all four boundaries.

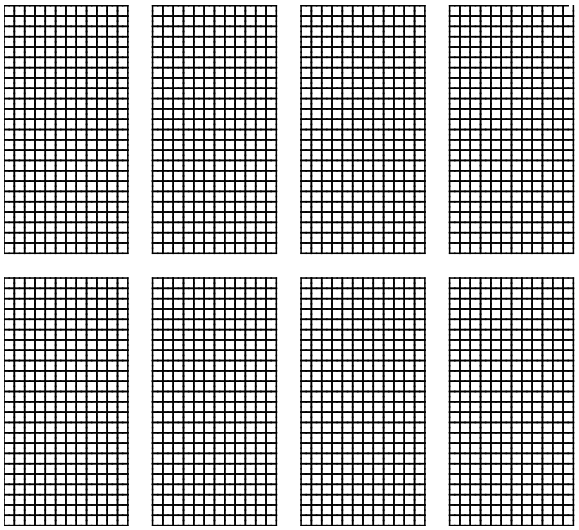


Figure 4.1: The mesh partition in both x and y directions for the square domain when 8 processes are used in parallel computing.

5.1 The Grid Refinement Study and the Polynomial Refinement Study

The L_2 error of density within the domain Ω at time t is used for the numerical performance studies of the CDFR scheme in this section. Specifically, it is defined as

$$e = \sqrt{\frac{\int_\Omega (\rho - \rho_{exact})^2 \, dV}{\int_\Omega 1 \, dV}}. \tag{5.3}$$

For the k-th order CDFR method the error can be estimated as

$$e \approx C \, \Delta x^k. \tag{5.4}$$

For the convenience of analyzing numerical performance, we further write (5.4) as

$$\log(e) = \log(C) + k \, \log(\Delta x). \tag{5.5}$$

For the CDFR method of certain order of accuracy k ($\log(C)$ is a constant for different mesh size Δx when k is fixed), if the error of the CDFR method satisfies the above estimate (5.4), $\log(e)$ should be linearly related to $\log(\Delta x)$. That is to say, when the mesh is refined from Δx_1 to Δx_2, the slope $(\log(e_1) - log(e_2))/(\log(\Delta x_1) - \log(\Delta x_2))$, which is the numerical order of accuracy, should be consistent with the theoretical value k. For a certain mesh size Δx, we expect the differences of $\log(C)$ among CDFR methods of different orders of accuracy to be trivial. Therefore, $\log(e)$ should be quasi-linearly related to k. Thus, when the order of accuracy is increased from k_1 to k_2, the slope $(\log(e_1) - \log(e_2))/(k_1 - k_2)$ should be around $\log(\Delta x)$, which means the error e drops exponentially as k increases. A uniform mesh set, i.e., $24 \times 24, 48 \times 48, 96 \times 96$ meshes, is used for the grid refinement study. The time step Δt is fixed as 1×10^{-4} to ensure that the numerical errors from time marching is negligible. The SSPRK3 scheme is used for time marching. Simulation ends at $t = 0.5$.

Figure 5.1 shows the density contours of the 3rd-order scheme on the 48×48 mesh. Figure 5.2a presents the numerical orders of accuracy results from the grid refinement study for the 3rd-, 4th-, and 5th-order schemes. In Figure 5.2a, the mesh size Δx and L_2 error e are both illustrated in log scale. The slope (the float number between two points in the figure) when one mesh is refined to another is the numerical order of accuracy. It is observed that the numerical orders of accuracy agree well with the theoretical values for all the tested CDFR methods in the grid refinement study. A polynomial refinement study of 3rd to 8th-order CDFR methods is also conducted on the 96×96 mesh. The result is shown in Figure 5.2b. In Figure 5.2b, the error is illustrated in log scale. For the polynomial refinement study, we do not quantitatively expect the slopes of $\log(e)$ versus the order of accuracy to be around $\log(\Delta x)$. In fact, we are simply intrested in if the error drops exponentially when the order of accuracy is increased. As shown in Figure 5.2b, over a wide range of polynomial refinement, when the order of accuracy increases, $\log(e)$ drops linearly. In other words, the convergence of the CDFR method family can be regarded as spectral.

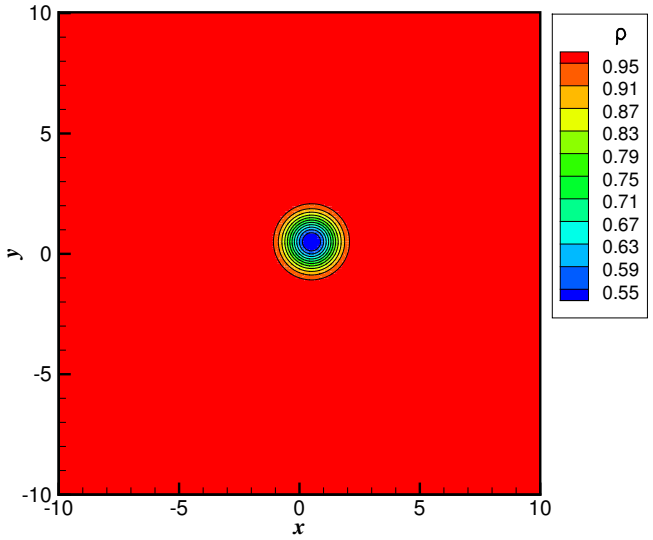
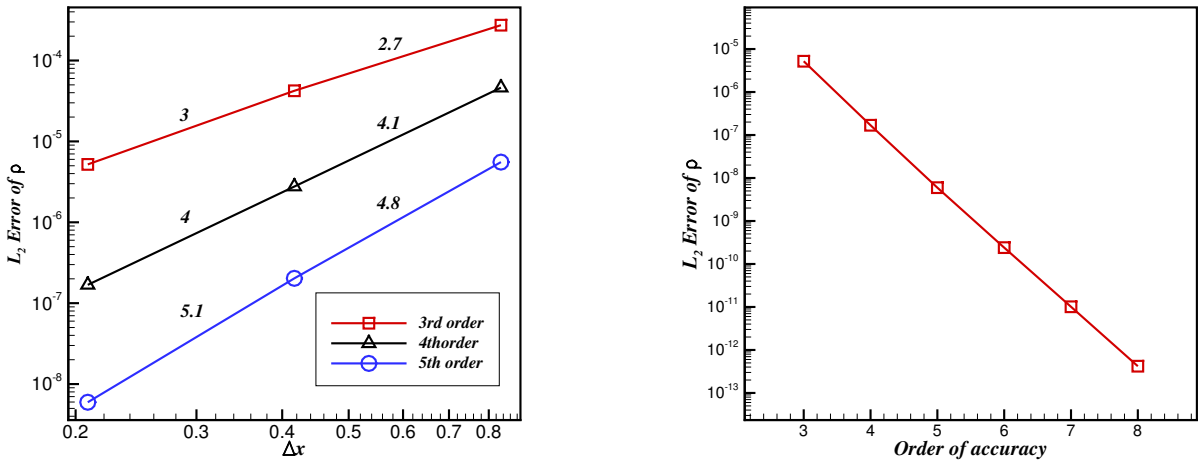


Figure 5.1: Density contour at $t = 0.5$ on the 48×48 mesh with the 5th order CDFR scheme for the isentropic Euler vortex propagation problem.



(a) The grid refinement study

(b) The polynomial refinement study

Figure 5.2: (a) The grid refinement study on a mesh set with $24 \times 24, 48 \times 48, 96 \times 96$ elements for the isentropic Euler vortex propagation problem using the 3rd to 5th order CDFR schemes and the floats in the figure are the numerical orders of accuracy when meshes are refined; (b) the polynomial refinement study for the 3rd to 8th order CDFR schemes on the 96×96 mesh.

5.2 The Parallel Performance Study

In this section, we focus on the parallel performance study of the CDFR method. The 3rd-order CDFR method is used in this section. We only let the simulations run 100 time steps with the time step size fixed as $\Delta t = 10^{-6}$ (small enough to make sure the Courant Friedrichs Lewy (CFL) condition satisfied for all the meshes). Different partition approaches have also been tested.

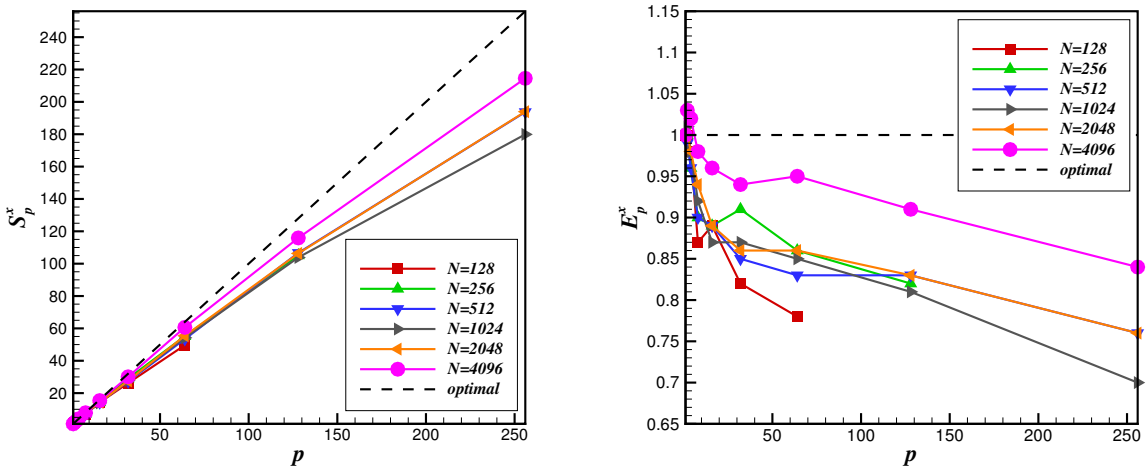
For the 3rd-order CDFR method, in each element we have $3 \times 3 \times (D + 2)$ degree of freedoms, where 3×3 is the number of Gauss points within the element and $D + 2$ is the number of variables on each Gauss point. For the 2D case, $D = 2$. In this case, the mesh is uniformly divided into $N \times N$ elements. So the total degree of freedom of the entire mesh can be calculated as $36N^2$.

Table 5.1: Memory prediction and observed memory usage for mesh partition in only x direction.

N	128	256	512	1024	2048	4096
Predicted	13.5 MB	54 MB	216 MB	864 MB	3.75 GB	13.5 GB
Observed	70 MB	239 MB	913 MB	3.5 GB	11 G	43 GB

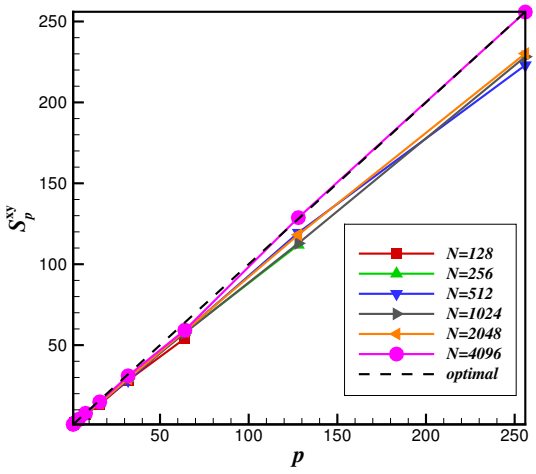
Table 5.2: The performance study of the 3rd-order CDFR scheme with SSPRK3 solving the isentropic vortex propagation in 100 time steps with $\Delta t = 10^{-6}$ on HPCF2013.

Mesh partition only in x direction									
(a) Wall clock time in HH:MM:SS									
N	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$	$p=32$	$p=64$	$p=128$	$p=256$
128	00:01:12	00:00:36	00:00:18	00:00:10	00:00:05	00:00:03	00:00:01	–	–
256	00:05:10	00:02:36	00:01:21	00:00:43	00:00:22	00:00:11	00:00:06	00:00:03	–
512	00:20:53	00:10:35	00:05:27	00:02:55	00:01:28	00:00:46	00:00:23	00:00:12	00:00:06
1024	01:24:46	00:41:57	00:21:41	00:11:28	00:06:04	00:03:03	00:01:34	00:00:49	00:00:28
2048	05:44:48	02:53:04	01:27:50	00:45:44	00:24:12	00:12:31	00:06:13	00:03:15	00:01:47
4096	25:04:41	12:13:14	06:07:01	03:11:11	01:37:45	00:50:03	00:24:49	00:12:59	00:07:01
(b) Observed S_p^x									
N	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$	$p=32$	$p=64$	$p=128$	$p=256$
128	1.00	1.99	3.93	6.97	14.31	26.17	49.64	–	–
256	1.00	1.99	3.82	7.21	14.28	28.98	55.18	105.12	–
512	1.00	1.97	3.83	7.18	14.27	27.22	53.40	106.66	193.71
1024	1.00	2.02	3.91	7.39	13.98	27.76	54.18	103.85	179.90
2048	1.00	1.99	3.93	7.54	14.25	27.55	55.42	106.35	193.94
4096	1.00	2.05	4.10	7.87	15.39	30.07	60.63	115.91	214.56
(c) Observed E_p^x									
N	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$	$p=32$	$p=64$	$p=128$	$p=256$
128	1.00	1.00	0.98	0.87	0.89	0.82	0.78	–	–
256	1.00	1.00	0.96	0.90	0.89	0.91	0.86	0.82	–
512	1.00	0.99	0.96	0.90	0.89	0.85	0.83	0.83	0.76
1024	1.00	1.01	0.98	0.92	0.87	0.87	0.85	0.81	0.70
2048	1.00	1.00	0.98	0.94	0.89	0.86	0.86	0.83	0.76
4096	1.00	1.03	1.02	0.98	0.96	0.94	0.95	0.91	0.84
Mesh partition in both x and y directions									
(c) Wall clock time in HH:MM:SS									
N	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$	$p=32$	$p=64$	$p=128$	$p=256$
128	00:01:12	00:00:36	00:00:19	00:00:10	00:00:05	00:00:03	00:00:01	–	–
256	00:05:10	00:02:36	00:01:16	00:00:43	00:00:22	00:00:11	00:00:05	00:00:03	–
512	00:20:53	00:10:35	00:05:04	00:02:51	00:01:26	00:00:45	00:00:21	00:00:10	00:00:06
1024	01:24:46	00:41:57	00:22:27	00:11:22	00:05:52	00:02:55	00:01:28	00:00:45	00:00:22
2048	05:44:48	02:53:04	01:29:48	00:44:24	00:24:19	00:11:47	00:05:53	00:02:55	00:01:30
4096	25:04:41	12:13:14	06:02:04	03:10:21	01:40:39	00:48:28	00:25:23	00:11:41	00:05:53
(d) Observed S_p^{xy}									
N	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$	$p=32$	$p=64$	$p=128$	$p=256$
128	1.00	1.99	3.72	7.34	13.33	28.23	54.12	–	–
256	1.00	1.99	4.07	7.16	13.98	29.37	57.86	111.95	–
512	1.00	1.97	4.12	7.31	14.51	28.09	58.48	119.47	223.01
1024	1.00	2.02	3.78	7.46	14.46	29.12	57.50	112.89	228.26
2048	1.00	1.99	3.84	7.77	14.18	29.27	58.57	118.19	230.20
4096	1.00	2.05	4.16	7.90	14.95	31.04	59.27	128.77	255.85
(e) Observed E_p^{xy}									
N	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$	$p=32$	$p=64$	$p=128$	$p=256$
128	1.00	1.00	0.93	0.92	0.83	0.88	0.85	–	–
256	1.00	1.00	1.01	0.89	0.87	0.92	0.90	0.87	–
512	1.00	0.99	1.03	0.91	0.91	0.88	0.91	0.93	0.87
1024	1.00	1.01	0.94	0.93	0.90	0.91	0.90	0.88	0.89
2048	1.00	1.00	0.96	0.97	0.89	0.91	0.92	0.92	0.90
4096	1.00	1.03	1.04	0.99	0.93	0.97	0.93	1.00	1.00

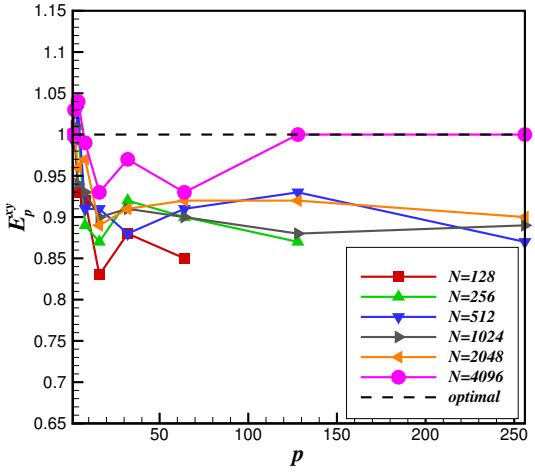


(a) Observed speedup S_p^x

(b) Observed efficiency E_p^x



(c) Observed speedup S_p^{xy}



(d) Observed efficiency E_p^{xy}

Figure 5.3: The performance study of the 3rd order CDFR scheme with SSPRK3 solving the isentropic vortex propagation in 100 time steps with $\Delta t = 10^{-6}$ on HPCF2013.

Recalling (3.14), we have also declared memory for $\boldsymbol{q}$ of the previous time step and $L(\boldsymbol{q})$. Thus, we can do a rough memory prediction here without including the memory usage of the auxiliary information for unstructured grids. The memory usage difference between different mesh partition approaches are not significant. Thus we only present the memory prediction and observed memory usage of mesh partition in only x direction, as illustrated in Table 5.1. It is discovered that the auxiliary information for the CDFR method on unstructured mesh consumes much larger memory space than the solutions on the solution points. The memory usage makes sense because for nodal high order methods on unstructured mesh, the geometry informations, such as connectivity, coordinates, coordinate transformations, etc., will consume a lot of memory.

The parallel scalability is often studied by the observed speedup S_p and efficiency E_p, where p is the number of processes. For a fixed size problem, the observed speedup S_p is defined as [11]

$$S_p = T_1/T_p, \tag{5.6}$$

where T_p denotes the time needed for p processes to solve the problem. And the observed efficiency E_p is defined as

$$E_p = S_p/p. \tag{5.7}$$

The performance study results are presented in Table 5.2 and Figure 5.3. Note that for $p = 1$, $E_p = S_p = 1$. In Table 5.2, the raw time, S_p and E_p of different partition approaches are illustrated. Herein we use supscript 'x' and

'xy' to distinguish the results of these two partition approaches, such as S_p^x and E_p^{xy}. Figure 5.3a and Figure 5.3b visualize the S_p^x and E_p^x while Figure 5.3c and Figure 5.3d visualize the S_p^{xy} and E_p^{xy} respectively. From the last row of raw wall clock time, it is observed the time for $N = 4096$ is over 25 hours for 100 time step size. This is why we do not conduct a performance study on a long physical time range. Generally, for both two partition approaches, it is observed that when the number of processes p increases , S_p and E_p will gradually decrease. And for bigger N, i.e., bigger problem size ($N > 512$), the speedup and efficiency will be better when more processes are used. Overall, scalability of the CDFR method is excellent, and the CDFR method can be efficiently accelerated utilizing parallel computing. Through the comparison between these two partition approaches, it is observed that dividing the mesh in both x and y directions the observed E_p^{xy} and S_p^{xy} will be better. Since the data need to be send between CPUs is less than only dividing the mesh in x direction when the numbers of elements on each process are the same.

6 Conclusions

In the present study, a brief introduction of the CDFR method is presented. We use the CDFR method as spatial discretization and SSPRK3 as the temporal discretization to solve the isentropic vortex propagation problem. Convergence rate studies have shown that the CDFR method performs well for conservation laws. The parallel performance study conducted in the present study has demonstrated that the CDFR method can benefit a lot from parallelization. Overall, the CDFR method is a compact high-order scheme family can be efficiently parallelized for high performance computing. Through the comparison of different mesh partition tests, it is also observed that by optimizing the communication cost between different processes the parallel performance can be obviously improved.

Acknowledgment

Wang and Yu gratefully acknowledge the support of the Office of Naval Research through the award N00014–16–1–2735, and the faculty startup support from the department of mechanical engineering at University of Maryland, Baltimore County (UMBC). The hardware used in the computational studies is part of the UMBC High Performance Computing Facility (HPCF). The facility is supported by the U.S. National Science Foundation through the MRI program (grant nos. CNS–0821258 and CNS–1228778) and the SCREMS program (grant no. DMS–0821311), with additional substantial support from the University of Maryland, Baltimore County (UMBC). See hpcf.umbc.edu for more information on HPCF and the projects using its resources.

References

[1] Z. J. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. T. Huynh, N. Kroll, High-order CFD methods: current status and perspective, International Journal for Numerical Methods in Fluids, vol. 72, issue 8, pp. 811–845, 2013.

[2] H. T. Huynh, A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods, in 18th AIAA Computational Fluid Dynamics Conference, AIAA–2007–4079, Miami, FL, 2007.

[3] H. T. Huynh, A reconstruction approach high-order schemes including discontinuous Galerkin methods for diffusion, in 47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace, AIAA–2009–403, Orlando, FL, 2009.

[4] H. T. Huynh, Z. J. Wang, P.E. Vincent, High-order methods for computational fluid dynamics: a brief review of compact differential formulations on unstructured grids, Computers & Fluids, vol. 98, pp. 209–220, 2014.

[5] J. Romero, K. Asthana, A. Jameson, A simplified formulation of the flux reconstruction method, Journal of Scientific Computing, vol. 67, issue 1, pp. 351–372, 2016.

[6] J. Romero, A. Jameson, Extension of the flux reconstruction method to triangular elements using collapsed-edge quadrilaterals, in 54th AIAA Aerospace Sciences Meeting, AIAA SciTech, AIAA 2016–1825, San Diego.

[7] L. Wang, M. L. Yu, Compact direct flux reconstruction for conservation laws, submitted.

[8] S. K. Lele, Compact finite difference schemes with spectral-like resolution, Journal of Computational Physics, vol. 103, issue 1, pp. 16–42, 1992.

[9] S. Gottlieb, C. W. Shu, E. Tadmor, Strong stability-preserving high-order time discretization methods, SIAM Review, vol. 43, issue 1, pp. 89–112, 2001.

[10] G. Karypis, V. Kumar, MeTiS—A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices—Version 4.0, University of Minnesota, Department of Computer Science and Engineering, Army HPC Research Center, Minneapolis, MN, 1998.

[11] S. Khuvis, M. K. Gobbert, Parallel performance studies for an elliptic test problem on the cluster maya, Technical Report HPCF–2015–6, UMBC High Performance Computing Facility, University of Maryland, Baltimore County, 2015.