18th AIAA Computational Fluid Dynamics Conference
25 - 28 June 2007, Miami, FL

AIAA 2007-4322

Efficient Implicit Non-linear LU-SGS Approach for Viscous Flow Computation Using High-Order Spectral Difference Method

Yuzhi Sun[1] and Z.J. Wang[2]
Department of Aerospace Engineering, Iowa State University, Ames, IA 50011

Yen Liu[3]
NASA Ames Research Center, Moffett Field, CA 94035

An implicit non-linear lower-upper symmetric Gauss-Seidel (LU-SGS) solution algorithm has been developed for a high-order spectral difference Navier-Stokes solver on unstructured hexahedral grids. The non-linear LU-SGS solver is preconditioned by the block element matrix, and the system of equations is then solved with a LU decomposition approach. The large sparse Jacobian matrix is computed numerically, resulting in extremely simple operations for arbitrarily complex residual operators. Several viscous test cases were performed to evaluate the performance. The implicit solver has shown speedup of 1 to 2 orders of magnitude over the multi-stage Runge-Kutta time integration scheme.

I. Introduction

Computational fluid dynamics (CFD) has undergone tremendous development as a discipline for three decades, and is used routinely to complement the wind tunnel in the design of aircraft. This has been made possible by progresses in many fronts, including numerical algorithms for the Navier-Stokes equations, grid generation and adaptation, turbulence modeling, flow visualization, as well as the dramatic increase in computer CPU and network speeds. Nearly all production flow solvers are based on second-order numerical methods. They are capable of delivering design-quality Reynolds Averaged Navier-Stokes (RANS) results with several million cells (degrees of freedom or DOFs) on commercial Beowulf clusters within a few hours.

As impressive as these second order codes are, there are still many flow problems considered out of reach, e.g., vortex dominated flows including helicopter blade vortex interaction, and flow over high-lift configurations. Unsteady propagating vortices are the main features of these flow problems, and second-order methods are too dissipative to resolve those unsteady vortices. The advantage of high-order methods (order of accuracy > 2) over first and second-order ones is well known in the CFD community. Generally speaking, with the same number of degrees-of-freedom (DOFs) or solution unknowns, high-order methods are capable of producing much more accurate results. For problems requiring very high accuracy, e.g., wave propagation problems in computational aeroacoustics, high-order methods have been the main choice. Many high-order methods were developed for structured grids, e.g., ENO/WENO methods[1,] compact methods[2-3], optimized methods[4], to name just a few. In the last two decades, there have been intensive research efforts on high-order methods for unstructured grids since many real world applications have complex geometries. An incomplete list of notable examples includes the spectral element method[5], multi-domain spectral method[6-7], k-exact finite volume method[8], WENO methods[9], discontinuous Galerkin (DG) method[10-12], high-order residual distribution methods[13], spectral volume (SV) [14-17] and spectral difference (SD) methods[18-21]. Among those methods, some are based on the weighted residual form of the governing equations, for instance the DG method[10-12]. Some are based on the integral form of the governing equations, e. g., the k-exact finite volume method[8] and SV methods[14-17]. Others, such as the staggered grid multi-domain spectral method[6-7] and the SD method[18-21], are based on the differential form.

When one chooses a particular method for three-dimensional applications, the cost and the complexity in implementing the method is often an important factor. It is obvious that methods based on the differential form are the easiest to implement since they do not involve surface or volume integrals. This is particularly true when high-

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

order curved boundaries need to be dealt with. We recently developed a high order SD method[22] for the three dimensional Navier-Stokes equations on unstructured hexahedral grids. High order accuracy and spectral convergence are achieved for several benchmark problems. It was also shown that the wall boundaries must be approximated with high-order surfaces. An explicit Runge-Kutta time integration scheme was used in the implementation. Although the explicit scheme is easy to implement and has high-order accuracy in time, it suffered from slow convergence, especially for viscous grids which are clustered in the viscous boundary layer. It is well-known that high-order methods are restricted to a smaller CFL number than low order ones. In addition, they also possess much less numerical dissipation. Therefore it takes excessive CPU to reach a state-steady solution with explicit high-order schemes. The computation cost of high-order explicit methods for many steady-state problems is so high that they become less efficient than low-order implicit methods in terms of the total CPU time given the same level of solution error. It is therefore imperative to develop efficient implicit solution approaches for high-order methods to fully realize the potentials, which is the objective of the present study.

Recently, a non-linear LU-SGS approach was successfully developed for an inviscid SD solver[23], and shown very promising convergence speedups of 1 to 2 orders. In this paper, we extend the approach to solve viscous flow problems governed by the Navier-Stokes equations. Many implicit schemes for unstructured grids were shown to be very effective for converging steady state flow problems[24-29]. The main difficulty in the extension is the computation of the element Jacobian matrices, which involve neighbor's neighbors. A numerical approach to compute the Jacobian is employed to avoid the difficulties.

The paper is organized as follows. In the next section, the formulation of the 3D spectral difference method including both explicit and implicit schemes is described for a hexahedral element. In Section 3, several representative test cases are selected to demonstrate the efficiency of the implicit approach. Conclusions and possible future works are summarized in Section 4.

II. Formulation of Multidomain Spectral Difference Method

Governing equation

Consider the unsteady compressible 3D Navier-Stokes equations in conservative form written as

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

where Q is the vector of conserved variables, and F, G, H are the total fluxes including both the inviscid and viscous flux vectors, i.e., $F=F^i-F^v$, $G=G^i-G^v$, $H=H^i-H^v$.

We employ non-overlapping unstructured hexahedral cells or elements to fill the computational domain. The use of hexahedral cells for viscous boundary layers is preferred over tetrahedral cells because of the efficiency and accuracy. In order to handle curved boundaries, both linear and quadratic isoparametric elements are employed, with linear elements used in the interior domain and quadratic elements near high-order curved boundaries. In order to achieve an efficient implementation, all elements are transformed from the physical domain (x,y,z) into a standard cubic element $(\xi,\eta,\varsigma)\in[0,1]\text{x}[0,1]\text{x}[0,1]$ as shown in Figure 1. The transformation can be written as

$$\begin{pmatrix}x\\y\\z\end{pmatrix}=\sum_{i=1}^{K}M_i(\xi,\eta,\varsigma)\begin{pmatrix}x_i\\y_i\\z_i\end{pmatrix}, \tag{2}$$

where K is the number of points used to define the physical element, (x_i,y_i,z_i) are the Cartesian coordinates of those points, and $M_i(\xi,\eta,\varsigma)$ are the shape functions. For the transformation given in (2), the Jacobian matrix J takes the following form

$$J=\frac{\partial(x,y,z)}{\partial(\xi,\eta,\zeta)}=\begin{bmatrix}x_\xi & x_\eta & x_\zeta\\ y_\xi & y_\eta & y_\zeta\\ z_\xi & z_\eta & z_\zeta\end{bmatrix}.$$

For a non-singular transformation, its inverse transformation must also exist, and the Jocobian matrices are related to each other according to

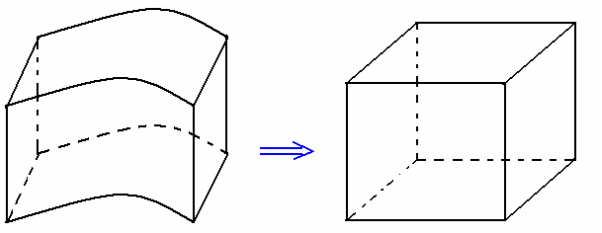


Figure 1. Transformation from a physical element to a standard element

$$\frac{\partial(\xi,\eta,\zeta)}{\partial(x,y,z)} = \begin{bmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{bmatrix} = J^{-1}.$$

The governing equations in the physical domain are then transformed into the computational domain (standard element), and the transformed equations take the following form

$$\frac{\partial \tilde{Q}}{\partial t} + \frac{\partial \tilde{F}}{\partial \xi} + \frac{\partial \tilde{G}}{\partial \eta} + \frac{\partial \tilde{H}}{\partial \varsigma} = 0. \tag{3}$$

where

$$\tilde{Q} = |J| \cdot Q$$

$$\begin{bmatrix} \tilde{F} \\ \tilde{G} \\ \tilde{H} \end{bmatrix} = |J| \begin{bmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{bmatrix} \cdot \begin{bmatrix} F \\ G \\ H \end{bmatrix}.$$

Let $\vec{S}_\xi = |J|(\xi_x,\xi_y,\xi_z)$, $\vec{S}_\eta = |J|(\eta_x,\eta_y,\eta_z)$, $\vec{S}_\zeta = |J|(\zeta_x,\zeta_y,\zeta_z)$. Then we have $\tilde{F} = \vec{f} \bullet \vec{S}_\xi$, $\tilde{G} = \vec{f} \bullet \vec{S}_\eta$, $\tilde{H} = \vec{f} \bullet \vec{S}_\zeta$ with $\vec{f} = (F,G,H)$.

Space Discretization

In the standard element, two sets of points are defined, namely the solution points and the flux points, illustrated in Figure 2 for a 2D element. The solution unknowns or degrees-of-freedom (DOFs) are the conserved variables at the solution points, while fluxes are computed at the flux points. In order to construct a degree *(N-1)* polynomial in each coordinate direction, solutions at *N* points are required. The solution points in 1D are chosen to be the Gauss points defined by

$$X_s = \frac{1}{2}\left[1 - \cos\left(\frac{2s-1}{2N} \cdot \pi\right)\right], \quad s = 1,2,\ldots,N. \tag{4}$$

The flux points are selected to be the Gauss-Lobatto points given by

$$X_{s+1/2} = \frac{1}{2}\left[1 - \cos\left(\frac{s}{N} \cdot \pi\right)\right], \quad s = 0,1,\ldots,N. \tag{5}$$

Using the *N* solutions at the solution points, a degree *N-1* polynomial can be built using the following Lagrange basis defined as

$$h_i(X) = \prod_{s=1, s\neq i}^{N} \left(\frac{X - X_s}{X_i - X_s}\right) \tag{6a}$$

Similarly, using the *N+1* fluxes at the flux points, a degree *N* polynomial can be built for the flux using a similar Lagrange basis defined as

$$l_{i+1/2}(X) = \prod_{s=0, s\neq i}^{N} \left(\frac{X - X_{s+1/2}}{X_{i+1/2} - X_{s+1/2}}\right). \tag{6b}$$

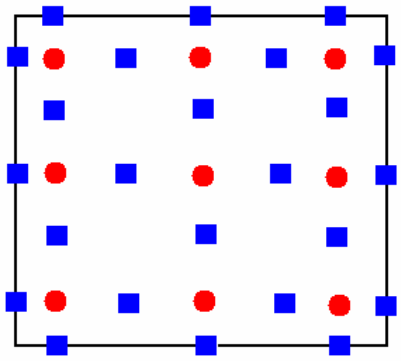


Figure 2. Distribution of solution points (circles) and flux points (squares) in a standard element for a 3rd order SD scheme.

The reconstructed solution for the conserved variables in the standard element is just the tensor products of the three one-dimensional polynomials, i.e.,

$$Q(\xi,\eta,\varsigma) = \sum_{k=1}^{N}\sum_{j=1}^{N}\sum_{i=1}^{N} \frac{\tilde{Q}_{i,j,k}}{|J_{i,j,k}|} h_i(\xi) \cdot h_j(\eta) \cdot h_k(\varsigma). \tag{7}$$

Similarly, the reconstructed flux polynomials take the following form:

$$\tilde{F}(\xi,\eta,\varsigma)=\sum_{k=1}^{N}\sum_{j=1}^{N}\sum_{i=0}^{N}\tilde{F}_{i+1/2,j,k}\; l_{i+1/2}(\xi)\cdot h_j(\eta)\cdot h_k(\varsigma)\,, \tag{8a}$$

$$\tilde{G}(\xi,\eta,\varsigma)=\sum_{k=1}^{N}\sum_{j=0}^{N}\sum_{i=1}^{N}\tilde{G}_{i,j+1/2,k}\; h_i(\xi)\cdot l_{j+1/2}(\eta)\cdot h_k(\varsigma)\,, \tag{8b}$$

$$\tilde{H}(\xi,\eta,\varsigma)=\sum_{k=0}^{N}\sum_{j=1}^{N}\sum_{i=1}^{N}\tilde{H}_{i,j,k+1/2}\; h_i(\xi)\cdot h_j(\eta)\cdot l_{k+1/2}(\varsigma)\,. \tag{8c}$$

The reconstructed fluxes are only element-wise continuous, but discontinuous across cell interfaces. For the inviscid flux, a Riemann solver, such as the Rusanov or Roe flux, is employed to compute a common flux at interfaces to ensure conservation and stability. In summary, the algorithm to compute the inviscid flux derivatives consists of the following steps:

- Given the conserved variables at the solution points $\left\{\tilde{Q}_{i,j,k}\right\}$, compute the conserved variables at the flux points $\{Q_{i+1/2,j,k}, Q_{i,j+1/2,k}, Q_{i,j,k+1/2}\}$ using (7) (Note that $h_m(X_n)=\delta_{mn}$);
- Compute the inviscid fluxes at the interior flux points using the solutions computed at Step 1, i.e., $\left\{\tilde{F}^i_{i+1/2,j,k}, i=1,\cdots,N-1\right\}$, $\left\{\tilde{G}^i_{i,j+1/2,k}, j=1,\cdots,N-1\right\}$, $\left\{\tilde{H}^i_{i,j,k+1/2}, k=1,\cdots,N-1\right\}$;
- Compute the inviscid flux at element interfaces using a Riemann solver, such as the Rusanov solver, in terms of the left and right conserved variables of the interface. Given the normal direction of the interface $\vec{n}$, and the averaged normal velocity component $\overline{V}_n$ and sound speed $\overline{c}$, the Rusanov flux on the interface is computed with

$$\tilde{F}^i=\frac{1}{2}(\tilde{F}^i_L+\tilde{F}^i_R-(\left|\overline{V}_n\right|+\overline{c})\cdot(Q_R-Q_L)\cdot\left|\vec{S}_\xi\right|\cdot sign\left(\vec{n}\bullet\vec{S}_\xi\right)$$

$$\tilde{G}^i=\frac{1}{2}(\tilde{G}^i_L+\tilde{G}^i_R-(\left|\overline{V}_n\right|+\overline{c})\cdot(Q_R-Q_L)\cdot\left|\vec{S}_\eta\right|\cdot sign\left(\vec{n}\bullet\vec{S}_\eta\right)$$

$$\tilde{H}^i=\frac{1}{2}(\tilde{H}^i_L+\tilde{H}^i_R-(\left|\overline{V}_n\right|+\overline{c})\cdot(Q_R-Q_L)\cdot\left|\vec{S}_\varsigma\right|\cdot sign\left(\vec{n}\bullet\vec{S}_\varsigma\right)$$

- Compute the derivatives of the fluxes at all the solution points according to

$$\left(\frac{\partial\tilde{F}}{\partial\xi}\right)_{i,j,k}=\sum_{r=0}^{N}\tilde{F}_{r+1/2,j,k}\cdot l'_{r+1/2}(\xi_i) \tag{9a}$$

$$\left(\frac{\partial\tilde{G}}{\partial\eta}\right)_{i,j,k}=\sum_{r=0}^{N}\tilde{G}_{i,r+1/2,k}\cdot l'_{r+1/2}(\eta_j) \tag{9b}$$

$$\left(\frac{\partial\tilde{H}}{\partial\varsigma}\right)_{i,j,k}=\sum_{r=0}^{N}\tilde{H}_{i,j,r+1/2}\cdot l'_{r+1/2}(\varsigma_k)\,. \tag{9c}$$

The viscous flux is a function of both the conserved variables and their gradients, i.e., $\tilde{F}^v_{i+1/2,j,k}=\tilde{F}^v(Q_{i+1/2,j,k},\nabla Q_{i+1/2,j,k})$. Therefore the key is how to compute the solution gradients at the flux points. The gradient of the conserved variables in the physical domain can be easily computed using

$$\nabla Q=\frac{\partial Q}{\partial\xi}\nabla\xi+\frac{\partial Q}{\partial\eta}\nabla\eta+\frac{\partial Q}{\partial\zeta}\nabla\zeta=\frac{1}{|J|}\left[\frac{\partial\left(Q\vec{S}_\xi\right)}{\partial\xi}+\frac{\partial\left(Q\vec{S}_\eta\right)}{\partial\eta}+\frac{\partial\left(Q\vec{S}_\varsigma\right)}{\partial\zeta}\right]. \tag{10}$$

In deriving (10), we have used the following identity

$$\frac{\partial\vec{S}_\xi}{\partial\xi}+\frac{\partial\vec{S}_\eta}{\partial\eta}+\frac{\partial\vec{S}_\varsigma}{\partial\zeta}=0.$$

The derivatives along each coordinate direction are computed using

$$\left[\frac{\partial\left(Q\vec{S}_{\xi}\right)}{\partial\xi}\right]_{j,k}=\sum_{r=0}^{N}\left(Q\vec{S}_{\xi}\right)_{r+1/2,j,k}\cdot l'_{r+1/2}(\xi) \tag{11a}$$

$$\left[\frac{\partial\left(Q\vec{S}_{\eta}\right)}{\partial\eta}\right]_{i,k}=\sum_{r=0}^{N}\left(Q\vec{S}_{\eta}\right)_{i,r+1/2,k}\cdot l'_{r+1/2}(\eta) \tag{11b}$$

$$\left[\frac{\partial\left(Q\vec{S}_{\varsigma}\right)}{\partial\zeta}\right]_{i,j}=\sum_{r=0}^{N}\left(Q\vec{S}_{\varsigma}\right)_{i,j,r+1/2}\cdot l'_{r+1/2}(\zeta)\,. \tag{11c}$$

The following steps are taken to compute the viscous fluxes:

- Same as Step 1 for the inviscid flux computations;
- When computing the derivatives using (11), the solution *Q* at the cell interface is not uniquely defined. The solution at the interface is simply the average of the left and right solutions,
$$\hat{Q}=(Q_L+Q_R)/2\,.$$
- Compute the gradients of the solution at the solution points using the solutions at the flux points with (10) and (11). Then the gradients are interpolated from the solution points to the flux points using the same Lagrangian interpolation approach given in (7).
- Compute the viscous flux at the flux points using the solutions and their gradients at the flux points. Again at cell interfaces, the gradients have two values, one from the left and one from the right. The gradients used in the viscous fluxes at the cell interface are simply the averaged ones, i.e.,
$$\tilde{F}^{v}=\tilde{F}^{v}((Q_L+Q_R)/2,(\nabla Q_L+\nabla Q_L)/2)\,.$$

Time Marching

Explicit scheme: Denote the residual at cell *c* $R_c(\tilde{Q}^n)$**.** Obviously , the semi-discrete equation can be written as

$$\frac{\partial\tilde{Q}_c}{\partial t}=R_c(\tilde{Q}^n)=-\left(\frac{\partial\tilde{F}}{\partial\xi}+\frac{\partial\tilde{G}}{\partial\eta}+\frac{\partial\tilde{H}}{\partial\varsigma}\right). \tag{12}$$

A multi-stage TVD Runge-Kutta scheme is used as the explicit scheme.

Implicit scheme: At each cell *c*, using the backward Euler difference, (1) can be written as

$$\frac{\tilde{Q}_c^{n+1}-\tilde{Q}_c^{n}}{\Delta t}-\left[R_c(\tilde{Q}^{n+1})-R_c(\tilde{Q}^{n})\right]=R_c(\tilde{Q}^{n}) \tag{13}$$

Let $\Delta\tilde{Q}_c=\tilde{Q}_c^{n+1}-\tilde{Q}_c^{n}$ and linearizing the residual, we obtain

$$R_c(\tilde{Q}^{n+1})-R_c(\tilde{Q}^{n})\approx\frac{\partial R_c}{\partial\tilde{Q}_c}\Delta\tilde{Q}_c+\sum_{nb\neq c}\frac{\partial R_c}{\partial Q_{nb}}\Delta\tilde{Q}_{nb}\,, \tag{14}$$

where *nb* indicates all the neighboring cells contributing to the residual of cell *c*. Therefore, the fully linearized equations for (3) can be written as

$$\left(\frac{I}{\Delta t}-\frac{\partial R_c}{\partial\tilde{Q}_c}\right)\Delta\tilde{Q}_c-\sum_{nb\neq c}\frac{\partial R_c}{\partial Q_{nb}}\Delta\tilde{Q}_{nb}=R_c(\tilde{Q}^{n})\,. \tag{15}$$

However, it costs too much memory to store the LHS implicit Jacobian matrices. Therefore, we employ a LU-SGS scheme to solve (15), i.e., we use the most recent solution for the nb cells,

$$\left(\frac{I}{\Delta t}-\frac{\partial R_c}{\partial\tilde{Q}_c}\right)\Delta\tilde{Q}_c^{(k+1)}=R_c(\tilde{Q}^{n})+\sum_{nb\neq c}\frac{\partial R_c}{\partial Q_{nb}}\Delta\tilde{Q}_{nb}^{*}\,. \tag{16}$$

The matrix

$$D = \left(\frac{I}{\Delta t} - \frac{\partial R_c}{\partial \tilde{Q}_c} \right) \tag{17}$$

is the element (or cell) matrix, which is not too large for 2nd to 4th order SD schemes. (16) is then solved with a direct LU decomposition solver. Since we do not want to store the matrices $\frac{\partial R_c}{\partial Q_{nb}}$, (16) is further manipulated as follows. Note that

$$\begin{aligned} & R_c(\tilde{Q}^n) + \sum_{nb \neq c} \frac{\partial R_c}{\partial \tilde{Q}_{nb}} \Delta \tilde{Q}^*_{nb} = R_c(\tilde{Q}^n_c, \{\tilde{Q}^n_{nb}\}) + \sum_{nb \neq c} \frac{\partial R_c}{\partial \tilde{Q}_{nb}} \Delta \tilde{Q}^*_{nb} \\ & \approx R_c(\tilde{Q}^n_c, \{\tilde{Q}^*_{nb}\}) \approx R_c(\tilde{Q}^*_c, \{\tilde{Q}^*_{nb}\}) - \frac{\partial R_c}{\partial \tilde{Q}_c} \Delta Q^*_c = R_c(\tilde{Q}^*) - \frac{\partial R_c}{\partial \tilde{Q}_c} \Delta Q^*_c. \end{aligned} \tag{18}$$

Combining (16) and (18) together, we obtain

$$\left(\frac{I}{\Delta t} - \frac{\partial R_c}{\partial \tilde{Q}_c} \right) (\tilde{Q}^{(k+1)}_c - \tilde{Q}^{(k)}_c) = R_c(\tilde{Q}^*) - \frac{\Delta Q^*_c}{\Delta t}. \tag{19}$$

Eq. (19) is then solved with symmetric forward and backward sweeps. Note that once (19) is solved to machine zero, the unsteady residual is zero at each time step. For steady state problems, the last term in (19) can be dropped sometimes resulting in faster convergence rate.

Computation of the Jacobian Matrix

Because of the way in which the viscous fluxes are computed, the present SD method uses cells which are neighbors' neighbors. If the analytical approach is used to compute the element Jacobian matrix $\frac{\partial R_c}{\partial \tilde{Q}_c}$, the formulation would be very complex. In stead, the following numerical approach is used based on the definition

$$\frac{\partial R_c}{\partial \tilde{Q}_c} \approx \frac{R_c(\{Q_{nb}\}, Q_c + \varepsilon) - R_c(\{Q_{nb}\}, Q_c)}{\varepsilon}. \tag{20}$$

where ε is a small parameter, e.g., $\varepsilon \approx \|Q_c\| \times 10^{-8}$. Although this approach is very easy to implement for arbitrarily complex residual operators, it is quite expensive because each variable has to be changed. In practice, we have found it is not necessary to compute the matrix at each iteration. Therefore, we often re-compute the matrix every 40-100 iterations. Numerical tests showed that this matrix-freezing approach did not significantly degrade the convergence rate to the steady state.

III. Numerical Experiments

Couette Flow

The Couette flow is a steady analytical solution of the Navier-Stokes equations, and was selected to demonstrate the performance with the implicit LU-SGS method. This problem models the viscous flow between a stationary, fixed temperature (T_0) front plate, and a moving, fixed temperature (T_1) rear plate at speed of *U*. The distance between the two plates is *H*. It has an exact solution under the simplification that the viscosity coefficient μ is a constant. The steady analytic solution is

$$u = \frac{U}{H} y, \quad v = 0, \quad w = 0$$

$$T = T_0 + \frac{y}{H}(T_1 - T_0) + \frac{\mu U^2}{2k} \cdot \frac{y}{H}\left(1 - \frac{y}{H}\right)$$

$$p = cons\tan t, \quad \rho = \frac{p}{R \cdot T},$$

where *k* is the thermal conductivity, and *R* is the gas constant.

In our simulations, we chose $U = 1.0, H = 2.0, T_0 = 0.8, T_1 = 0.85, \mu = 0.01$. The computational domain is [0,4]x[0,2]x[0,4]. We conducted an efficiency study of implicit LU-SGS method using 2nd to 4th order schemes on a coarse grid with 16 (4x4x1) elements. The residual histories in terms of CPU time for 2nd to 4th order SD schemes are compared with the explicit schemes in Figure 3. The implicit schemes showed over an order of magnitude speedup. The residual histories in terms of iteration numbers and CPU times using the implicit schemes are plotted in Figure 4. It is interesting to note that the convergence is nearly independent of the order of accuracy. Obviously, it takes much more CPU time to converge the higher order schemes.

Steady viscous flow around a sphere

A steady viscous flow around a sphere is used here to demonstrate the performance with the implicit LU-SGS method. The mesh used is shown in Figure 5. The Reynolds number based on the diameter was chosen to be 118 since an experimental streamline picture is available for comparison. This viscous flow computation was performed using the 2nd to 4th order schemes. The computational streamlines agree well with experimental streamlines. The explicit and implicit schemes are compared in Figure 6, which shows the convergence histories in terms of CPU times. For all the tested schemes, the speedup with the LU-SGS algorithm is more than an order of magnitude and up to 2 orders, fully demonstrating the effectiveness of the implicit algorithm. The implicit SD schemes of various orders of accuracy are also compared in Figure 7. Note that the convergence in terms of iterations is nearly order independent. It is also obviously the CPU times for higher order schemes increase non-linearly with respect to the order of accuracy.

Laminar flow over NACA0012 airfoil

In this test, we consider a subsonic viscous flow problem over the NACA0012 airfoil at an angle of attack $\alpha = 0^0$, free stream Mach number $M = 0.5$, and Reynolds number $\mathrm{Re} = 5000$. This is a widely used validation case for viscous flow solvers. The computational grid is displayed in Figure 8 with size . The larger number refers to the number of cells distributed along the airfoil surface and the smaller one the number of cells in the radial and span-wise direction. The grid extents about 20 chords away form the airfoil. The computations were performed using 2nd to 4th order SD schemes. The wall is assumed to be non-slip and adiabatic, and is represented by a quadratic patch. The Reynolds number is near the upper limit for a steady laminar flow. A distinguishing feature of this test case is the separation region of the flow occurring near the trailing edge, which causes the formation of a small recirculation bubble that extends in the near-wake region of the airfoil.

Figure 9 shows the Mach contours computed with SD schemes of 3rd and 4th order of accuracy. It is obvious that the solution is getting smoother and smoother with the increasing of the order of polynomial reconstruction, indicating the solution is more accurate. The convergence history of explicit and implicit 3rd SD schemes is plotted in Figure 10. The speedup factor in this case is estimated to be more than 2 orders. The residual histories of various implicit schemes are plotted in Figure 11. Note that in this case, the convergence rate in terms of iterations is strongly dependent on the order of accuracy. It is speculated that this is due to the high aspect ratio cells used in the computational grid.

IV. Conclusions

In this paper, an efficient implicit lower-upper symmetric Gauss-Seidel (LU-SGS) solution algorithm has been extended to viscous flow simulations using a high order multi-domain spectral difference method on unstructured hexahedral grids. A numerical approach is developed to compute the element Jacobian matrix, resulting in straightforward operations for arbitrarily complex residual operators. The implicit scheme has shown 1 to 2 orders of magnitude of speed-up relative to the multi-stage Runge-Kutta explicit time integration scheme for several demonstration problems..

Acknowledgements

The study has been funded by Rockwell Scientific/DARPA under contract W911NF-04-C-0102, and partially supported by the Air Force Office of Scientific Research. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, AFOSR, or the U.S. Government.

References

[1]Shu, C.-W., Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws, in Advanced Numerical Approximation of Nonlinear Hyperbolic Equations, edited by A. Quarteroni, Lecture Notes in Mathematics (Springer-Verlag, Berlin/New York, 1998), Vol. 1697, P. 325.

[2]Lele, S.K., "Compact Finite Difference Schemes with Spectral-Like Resolution," Journal of Computational Physics, Vol. 103, 1992, pp. 16-42.

[3]Visbal, M. and Gaitonde, D., Shock Capturing Using Compact- Differencing- Based Methods, AIAA-2005-1265.

[4]Tam, C.K.W., and Webb, J.C., "Dispersion-Relation-Preserving Finite difference Schemes for Computational Acoustics," Journal of Computational Physics, Vol. 107, No. 2, 1993, pp. 262-281.

[5]Patera, A.T., A Spectral element method for fluid dynamics: Laminar flow in a channel expansion, J. Comput. Phys. 54, 468(1984).

[6]Kopriva, D.A. and Kolias, J.H., A conservative staggered –grid Chebyshev multidomain method for compressible flows, J. Comput. Phys. 125, 244(1996).

[7]Kopriva, D.A., A Staggered-Grid Multidomain Spectral Method for the Compressible Navier–Stokes Equations, Journal of Computational Physics, Volume 143, pp. 125-158, 1998.

[8]Barth, T.J. and Frederickson, P.O., High-Order Solution of the Euler Equations on Unstructured Grids using Quadratic Reconstruction, AIAA Paper No. 90-0013(1990).

[9]Hu, C. and Shu, C.-W., Weighted essentially non-oscillatory schemes on triangular meshes, J. Comput. Phys. 150, 97 (1999).

[10]Cockburn, B. and Shu, C.-W., TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework, Math. Comput. 52,411(1989).

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

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

[13]Abgrall, R. and Roe, P.L., High Order Fluctuation Schemes on Triangular Meshes, Journal of Scientific Computing, Volume 19, pp. 3 – 36, 2003.

[14]Wang, Z.J. and Liu, Yen, Spectral (Finite) Volume Method for Conservation Laws on Unstructured Grids III: Extension to One-Dimensional Systems, J. Scientific Computing, Vol. 20 No. 1, pp.137-157 (2004).

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

[16]Liu, Y., Vinokur, M., and Wang, Z.J., Spectral (Finite) Volume Method for Conservation Laws on Unstructured Grids V: Extension to Three-Dimensional Systems, Journal of Computational Physics Vol. 212, pp. 454-472 (2006).

[17]Sun, Yuzhi, Wang, Z.J., and Liu, Yen, Spectral (Finite) Volume Method for Conservation Laws on Unstructured Grids VI: Extension to Viscous Flow, Journal of Computational Physics, Vol. 215,pp.41-58 (2006).

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

[19]Liu, Yen, Vinokur, M., and Wang, Z.J., Multi-Dimensional Spectral Difference Method for Unstructured Grids, AIAA-2005-0320.

[20]Wang, Z. J., and Liu, Yen, The Spectral Difference Method for the 2D Euler Equations on Unstructured Grids, AIAA-2005-5112.

[21]Huang, P.G., Wang, Z.J., and Liu, Yen, An Implicit Space-Time Spectral Difference Method for Discontinuity Capturing Using Adaptive Polynomials, AIAA-2005-5255.

[22]Sun, Yuzhi, Wang, Z.J., and Liu, Yen, High-Order Multi-domain Spectral Difference Method for the Navier-Stokes Equations on Unstructured Hexahedral Grids, *Communications in Computational Physics* Vol. 2, No. 2, pp. 310-333 (2007), and also AIAA-2006-301.

[23]Sun, Yuzhi, Wang, Z.J., Liu, Y. and Chen C.L., Efficient Implicit LU-SGS Algorithm for High-Order Spectral Difference Method on Unstructured Hexahedral Grids, AIAA-2007-0313.

[24]Venkatakrishnan, V., and Mavriplis, D.J., Implicit Solvers for Unstructured Meshes, Journal of Computational Physics, Vol. 105, No. 1, 1993, pp.83-91.

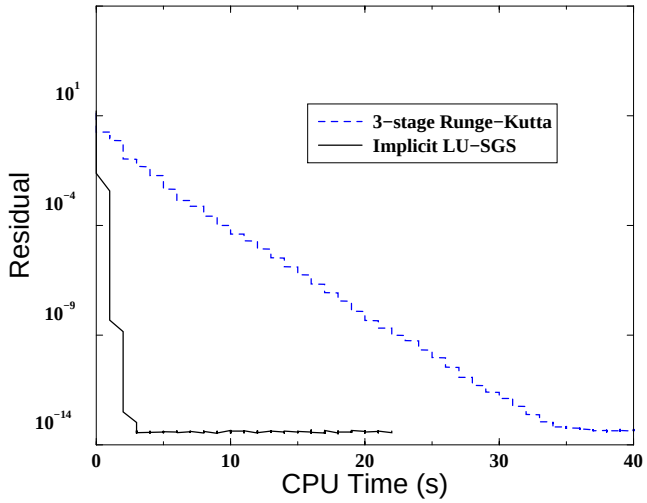
[25]Venkatakrishnan, V., and Mavriplis, D.J., Implicit Method for the Computation of Unsteady Flows on Unstructured Grids, Journal of Computational Physics, Vol. 127, No.2, 1996, pp.380-397.

[26]Soetrisno, M., Imlay, S.T. and Roberts, D.W., A zonal Implicit Procedure for Hybrid Structured-Unstructured Grids, AIAA Paper 94-0645, Jan. 1994.

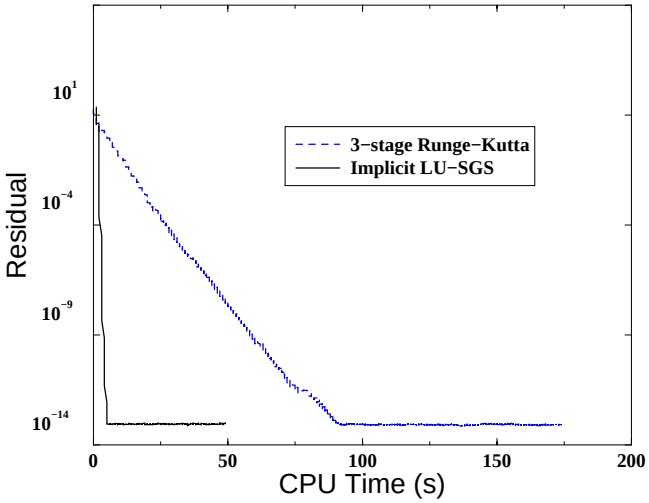
[27]Frink, N.T., Assessment of an Unstructured-Grid Method for Predicting 3-D Turbulent Viscous Flows, AIAA Paper 96-0292, Jan. 1996.

[28]Blanco, M., and Zingg, D.W., A Fast Solver for the Euler Equation on Unstructured Grids Using a Newton-GMRES Metod, AIAA Paper 97-0331, Jan. 1997.

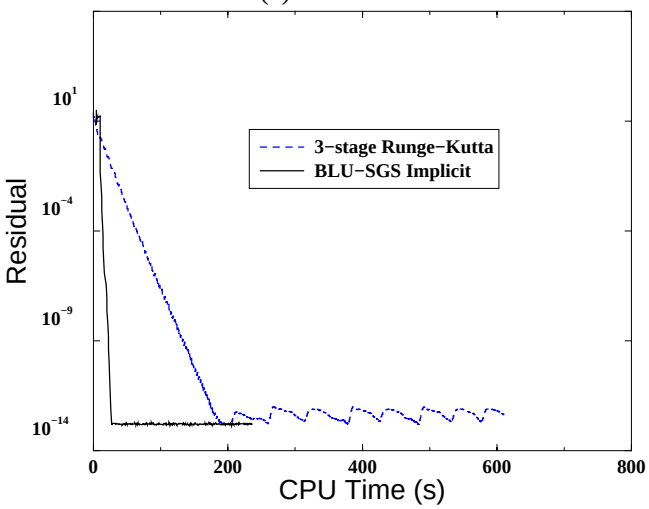
[29]Chen, R.F. and Wang, Z.J., Fast, Block Lower-Upper Symmetric Gauss-Seidel Scheme for Arbitrary Grids.



(a) 2nd order



(b) 3rd order



(c) 4th order

Figure 3. Residual history in term of CPU Time for Couette flow

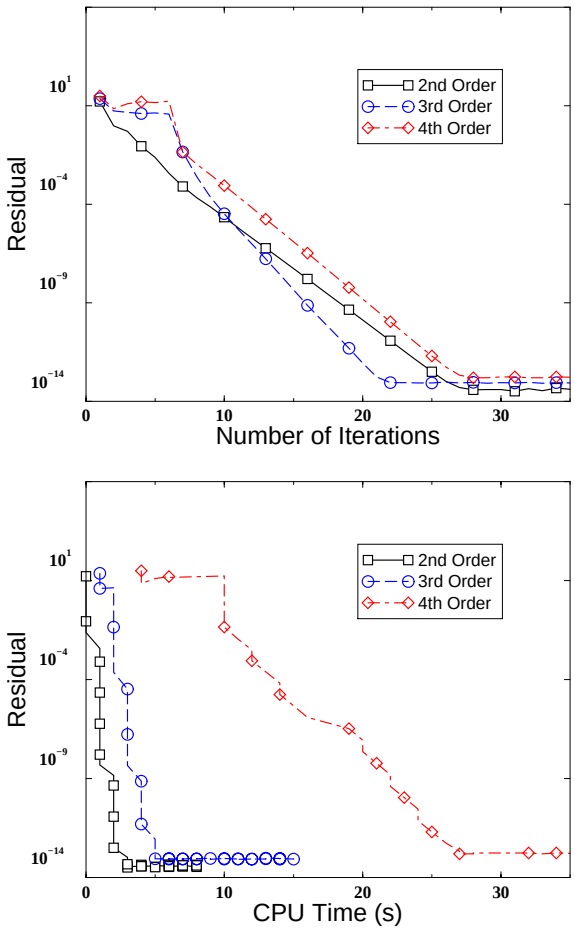


Figure 4. Convergence histories for various SD schemes on the same mesh

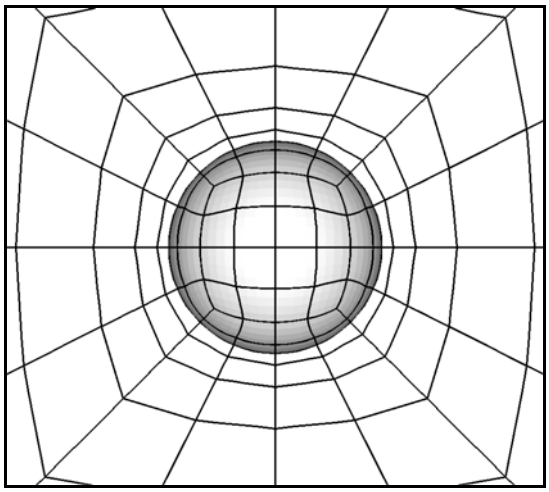
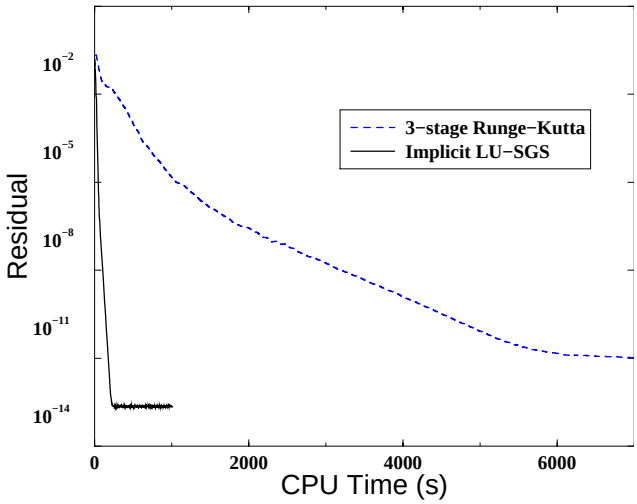
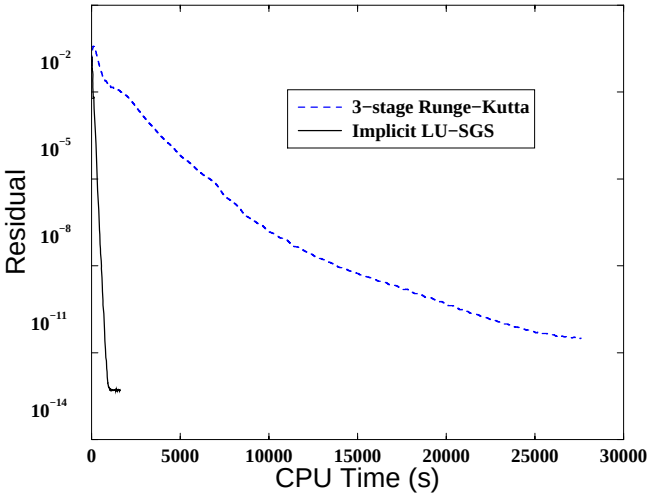


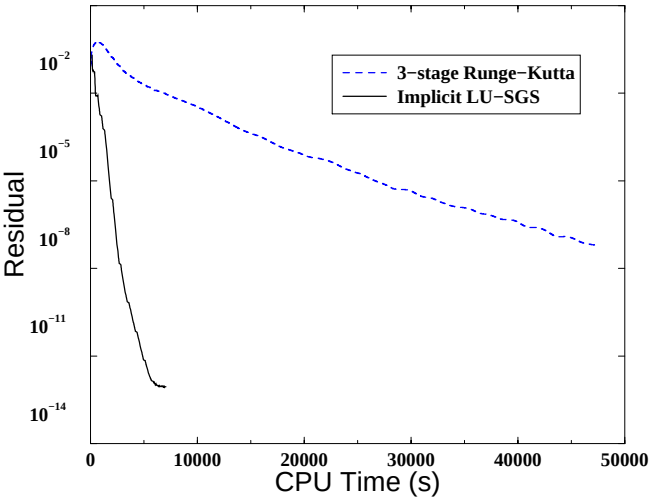
Figure 5. Sphere grids with quadratic boundary (768 elements)



(a) 2nd order



(b) 3rd order



(c) 4th order

Figure 6. Residual histories in term of CPU times for viscous flow over a sphere

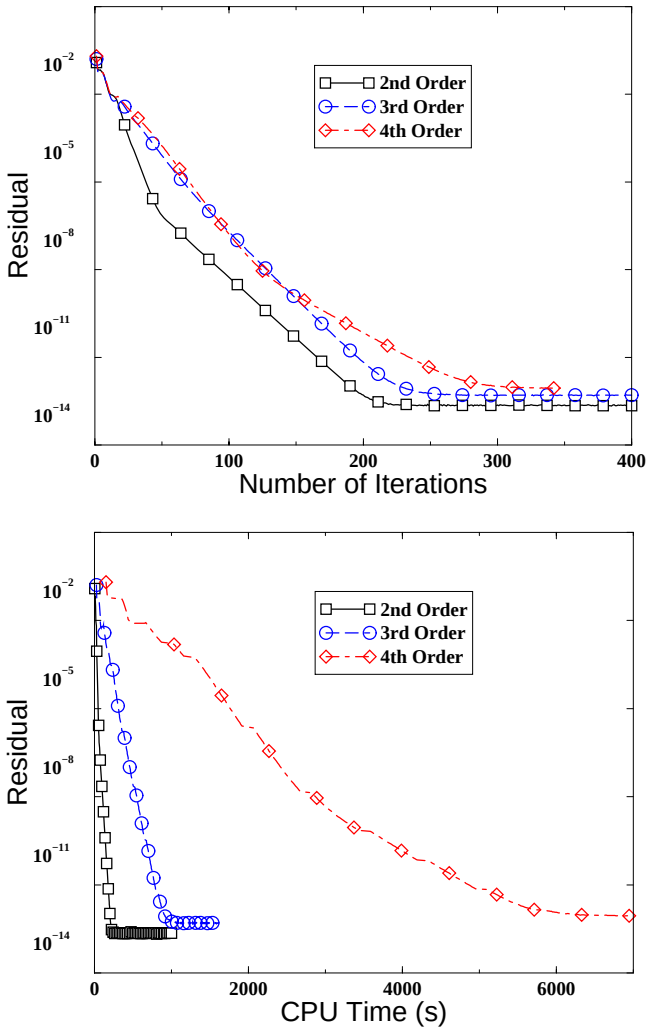


Figure 7. Convergence histories in terms of CPU times for various SD schemes

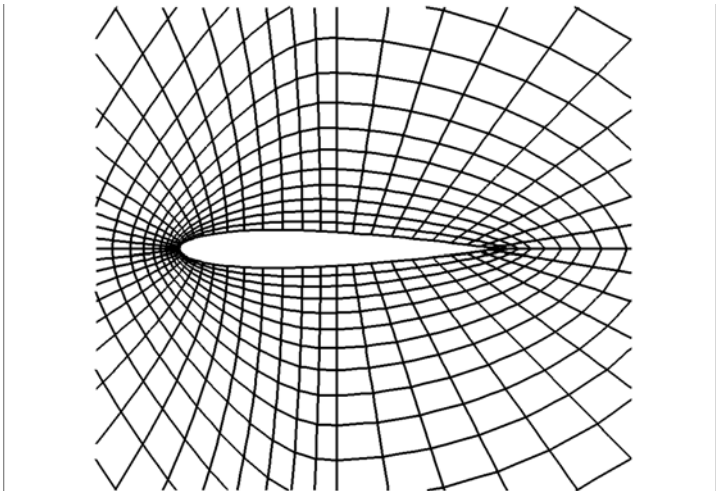
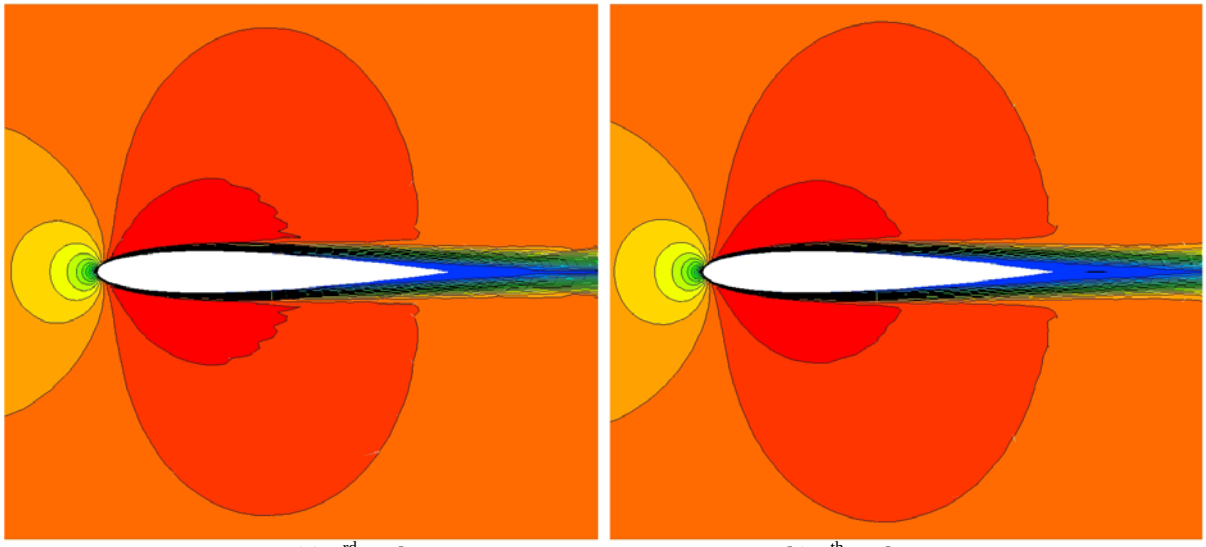


Figure 8. Computational grid around a NACA0012 airfoil



(a) 3rd Order (b) 4th Order

Figure 9. Computed Mach number contours using 3rd and 4th order SD schemes.

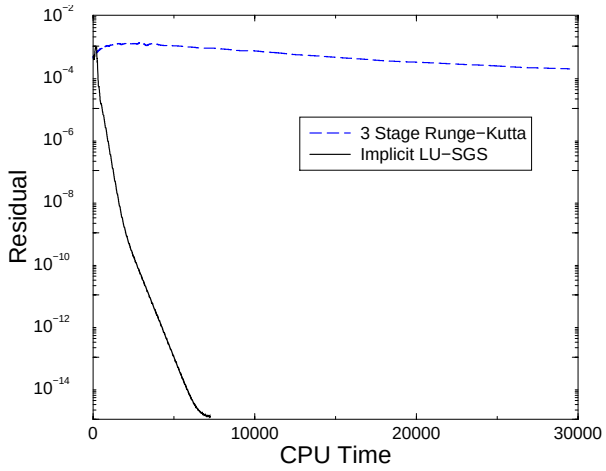


Figure 10. Convergence histories for 3rd order SD scheme

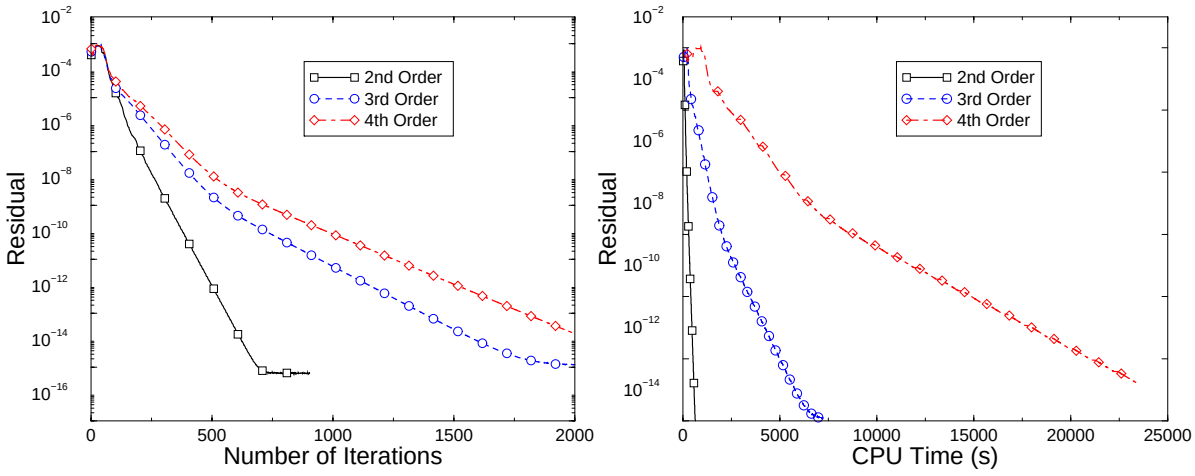


Figure 11. Convergence histories in terms of CPU times for laminar flow over the NACA0012 Airfoil