Implicit scheme for meshless compressible Euler solver

Manish K. Singh\textsuperscript{a,b,*}, V. Ramesh\textsuperscript{b} and N. Balakrishnan\textsuperscript{b}

\textsuperscript{a}CTFD Division, CSIR-NAL, Bangalore, India; \textsuperscript{b}Aerospace Engineering Department, Indian Institute of Science, Bangalore, India

(Received 3 March 2014; final version received 3 May 2015)

In this paper, an implicit scheme is presented for a meshless compressible Euler solver based on the Least Square Kinetic Upwind Method (LSKUM). The Jameson and Yoon’s split flux Jacobians formulation is very popular in finite volume methodology, which leads to a scalar diagonal dominant matrix for an efficient implicit procedure (Jameson & Yoon, 1987). However, this approach leads to a block diagonal matrix when applied to the LSKUM meshless method. The above split flux Jacobian formulation, along with a matrix-free approach, has been adopted to obtain a diagonally dominant, robust and cheap implicit time integration scheme. The efficacy of the scheme is demonstrated by computing 2D flow past a NACA 0012 airfoil under subsonic, transonic and supersonic flow conditions. The results obtained are compared with available experiments and other reliable computational fluid dynamics (CFD) results. The present implicit formulation shows good convergence acceleration over the RK4 explicit procedure. Further, the accuracy and robustness of the scheme in 3D is demonstrated by computing the flow past an ONERA M6 wing and a clipped delta wing with aileron deflection. The computed results show good agreement with wind tunnel experiments and other CFD computations.

\textbf{Keywords:} implicit scheme; meshless Euler solver; LSKUM; matrix-free formulation; NACA 0012 airfoil; ONERA M6 wing; clipped delta wind; aileron deflection

1. Introduction

In the past decade a lot of work has been done towards the development of meshless solvers. The basic idea of such a solver is to develop a method which does not require the process of breaking up the physical domain in terms of smaller hexahedral cells (structured grids) or tetrahedral cells (unstructured grids). The meshless solver only needs a cloud of points to discretize the governing equations. There is no requirement for points to connect in order to form the grids, which are used in the conventional finite volume and finite element methods. This type of solver is able to operate on any arbitrary distribution of points, thus mitigating the problem of generating an entire complex grid. For a complex configuration, with such a solver, we can generate a cloud of points over the entire geometry with relative ease. Individual clouds are generated over each component independently and then merged to form a single large cloud. A suitable deletion operation is carried out to remove hidden points.

One of the earliest meshless methods is called Smoothed Particle Hydrodynamics (SPH; Gingold & Monaghan, 1977; Lucy, 1977). This method was originally developed for astrophysical applications and later used by many researchers for problems in solid mechanics and fluid mechanics (Monaco et al., 2011; Pu, Shao, Huang, & Hussain, 2013). Liszka has developed a generalized finite difference method (meshless) which uses the moving least square approach for derivative approximation (Demkowicz, Karafiát, & Liszka, 1984; Liszka & Orkisz, 1980). The early work on the use of meshless methods for fluid dynamics applications can be traced back to the work of Deshpande (1986a, 1986b) and Batina (1993). Both of them have used the least square procedure to approximate spatial derivatives. Deshpande and colleagues have developed a meshless Euler solver called the Least Square Kinetic Upwind Method (LSKUM; Deshpande, Anandhanarayanan, Praveen, & Ramesh, 2002; Deshpande, Kulkarni, & Ghosh, 1998; Ghosh & Deshpande, 1995; Ramesh, 2001; Ramesh & Deshpande, 2001, 2004, 2007; Ramesh, Mathur, & Deshpande, 1997). Sridar and Balakrishnan (2003, 2006) have developed a meshless solver based on the finite difference approach called the Least Square Upwind Finite Difference Method (LSFD-U). In this approach, the mid-point between a given node and its neighbor is used to determine the upwind fluxes (Balakrishnan & Sridar, 2001). This scheme has the flexibility to choose between different flux formulations such as the Roe, Van Leer, Kinetic Flux Vector Split (KFVS) and the Advection Upstream Splitting Method (AUSM). The capabilities of the LSFD-U were further enhanced by including viscous terms (Munikrishna, 2007; Munikrishna & Balakrishnan, 2007; Ninawe, Munikrishna, & Balakrishnan, 2006). Katz and Jameson (2009a, 2009b, 2009c) have developed an edge-based meshless Euler solver called the ‘meshless volume scheme’. They have demonstrated this method along with a multi-grid convergence acceleration
technique, which they named the ‘multi-cloud approach’, using the NACA 0012 and RAE 2822 airfoils. Zhau and Xu (2010) used the meshless method to compute an unsteady flow field over the NACA 0012 and NACA 0008 airfoils. Mahindra, Singh, and Gouthaman (2011) used an LSKUM-based solver to compute the viscous rotating flow. They observed the ill-conditioning of a least square problem when an explicit LSKUM solver operates on the stretched distribution of points near a boundary, which is required to capture viscous flow features. They chose split stencils (a set of neighbors) selectively to avoid the ill-conditioning of the least square procedure. The present work is focused on in-viscid flow, which does not require the stretched distribution of points as required for viscous solvers. Such difficulties are more pronounced for explicit scheme-based solvers. Moreover, implicit schemes are more stable than explicit schemes.

It is difficult to show mathematically that the above meshless schemes are formally conservative. However, it is shown through numerical experiments that these schemes have mass conservation properties (Sridar & Balakrishnan, 2003). Recently, Chiu, Wang, and Jameson (2011) have developed a finite volume-like conservative meshless scheme in which the discrete coefficients are determined by an algorithm based on minimum-norm solutions such that the scheme is locally conservative.

Many of the above solvers have used explicit time stepping. Generally, computational time per iteration for meshless solvers is more than their finite volume counterpart. So there is interest in convergence acceleration techniques such as implicit methodology or the multi-grid approach. Katz and Jameson (2009c) have reported a convergence acceleration of about 20 using a multi-cloud approach on the NACA 0012 airfoil. The implicit Lower-Upper Symmetric Gauss-Seidel (LU-SGS) scheme is very attractive for convergence acceleration due to its low numerical complexity, moderate memory requirement and unconditional stability for linear wave equations. LU-SGS has been used widely for many Finite Volume Method-based solvers (Shende, 2005; Shende & Balakrishnan, 2004). Sridar and Balakrishnan (2006) have implemented a LU-SGS implicit procedure along with Symmetric Gauss-Seidel (SGS) and modified SGS (SGS-M) implicit procedures on the finite difference-based meshless Euler solver LSFD-U. These implicit procedures were tested on subsonic and transonic flow over a NACA 0012 airfoil, supersonic flow past the ramp in channel and high supersonic flow past the half cylinder. It was found that the LU-SGS implicit procedure performs better than the SGS implicit procedure for flow past the NACA 0012 airfoil (both subsonic and transonic) and supersonic flow past the ramp in channel. The modified SGS procedure gives the best performance. For the modified SGS implicit procedure, a convergence speed-up of at least 20 is observed as compared to the four-stage Range-Kutta (RK4) explicit scheme. Anandhanarayanan (2003) implemented a LU-SGS implicit procedure in a LSKUM meshless Euler solver and obtained a maximum convergence speed close to 4, on flow over bump.

The implicit version of flux vector splitting schemes requires the computation of split flux Jacobians, which is expensive. To avoid this, a matrix-free approach is implemented in finite volume methods for off diagonal terms of the implicit operator (Luo, Baum, & Lohner, 1998). The split flux Jacobians are very popular for diagonal terms with regard to retaining diagonal dominancy (Jameson & Yoon, 1987; Yoon & Jameson, 1986, 1987). In the finite volume or finite difference method this leads to a scalar diagonal matrix. However, this procedure leads to a block diagonal matrix, which again requires expensive inversion when applied to a meshless solver. Anandhanarayanan (2003) has approximated a split flux Jacobian to be half of the spectral radius in order to avoid a block diagonal matrix. Conventional LSKUM solvers achieve second-order accuracy via a two-step defect-correction procedure. The present solver uses modified Courant-Isaacson-Rees (CIR) splitting. We have called this an LSKUM-MCIR or simply MCIR scheme. This enables us to get a single-step higher-order meshless method. It is in this respect that the present solver is distinctly different from the work of Mahindra et al. (2011) and Anandhanarayanan, Nagarathinam, and Deshpande (2004).

In the present work we have developed a modified implicit procedure using a single-step higher-order scheme which can be considered an extension of earlier work on LSKUM-based solvers. We use Jameson and Yoon’s split flux Jacobian formulation for diagonal terms in order to retain diagonal dominancy for an efficient implicit procedure. However, a matrix-free approach is applied for diagonal as well as off-diagonal terms of the implicit operator as against the finite volume or finite difference method where it is required to be applied for off-diagonal terms only. This modification leads to a scalar diagonal matrix using a meshless method. The present implicit procedure is tested on a NACA 0012 airfoil, an ONERA M6 wing and a clipped delta wing with aileron deflection. The results are compared with those of the RK4 explicit meshless method and available wind tunnel experiments.

The paper outline is as follows. The LSKUM is presented in section 2. Time integration schemes are presented in section 3 with special emphasis on the adaptation aspect of the LU-SGS implicit scheme for a meshless solver. 2D and 3D results are presented in section 4, followed by the conclusion in section 5.

2. Least Square Kinetic Upwind Method (LSKUM)
2.1. Formulation

The Boltzmann equation in two dimensions for a Maxwellian velocity distribution function $F$ (Deshpande,
The approximations for spatial derivatives in equation (3) for split fluxes $G^{x+}$, $G^{x-}$, $G^{y+}$ and $G^{y-}$ are obtained by the least square procedure and using the split stencils $x^+, x^-, y^+$ and $y^-$ respectively. The split stencils $x^+, x^-, y^+$ and $y^-$ are shown as shaded areas in Figures 1(a–d), respectively. Let ‘$i$’ be the point at which we want to calculate the spatial derivative and point ‘$k$’ be a point in the neighborhood of point ‘$i$’. Then the approximation for the spatial derivatives at point ‘$i$’ is given by

$$\left[ \sum \frac{\Delta x_k^2}{\Delta x_k} \sum \frac{\Delta x_k \Delta y_k}{\Delta y_k} \right] F_{x_i} = \left[ \sum \frac{\Delta x_k \Delta y_k}{\Delta y_k} \sum \frac{\Delta x_k^2}{\Delta x_k} \right] F_{y_i} \right].$$

(4)

The space derivative approximation given by equation (4) is first-order accurate. Second-order accuracy can be obtained by using the two-step defect-correction method of Ghosh and Deshpande (1995). But, in the present work, the single-step MCIR scheme proposed by Ramesh and Deshpande (2004) has been used to obtain higher-order approximations for the spatial derivative. In this scheme, molecular velocity is written as

$$v_1 = \frac{v_1 + |v_1|\varphi_1 + v_1 - |v_1|\varphi_1}{2},$$

$$v_2 = \frac{v_2 + |v_2|\varphi_2 + v_2 - |v_2|\varphi_2}{2},$$

then the Boltzmann equation can be written as

$$\frac{\partial F}{\partial t} + \frac{v_1 + |v_1|\varphi_1}{2} \frac{\partial F}{\partial x} + \frac{v_1 - |v_1|\varphi_1}{2} \frac{\partial F}{\partial x} + \frac{v_2 + |v_2|\varphi_2}{2} \frac{\partial F}{\partial y} + \frac{v_2 - |v_2|\varphi_2}{2} \frac{\partial F}{\partial y} = 0,$$

(5)

where $\varphi_1$ and $\varphi_2$ represents the dissipation control parameters corresponding to two components of the molecular velocity. Now, several choices are possible for $\varphi_1$ and $\varphi_2$. Dominic (2004) had chosen dissipation control parameters such that modified split-fluxes satisfy the Rankine-Hugoniot condition across the shock. Anil (2008) has taken the dissipation control parameter as a function of molecular velocity. In the present work, the dissipation control parameters $\varphi_1$ and $\varphi_2$ are chosen as $\varphi_1, \varphi_2 = (\Delta r)^p$, where $0 < p < 1$ and $\Delta r$ is the distance between any point and its nearest neighborhood. In numerical experiments it has been observed that when $p$ reaches close to 1, wiggles appear in the solution. The optimum value of $p$ was found to be $.25 < p < .7$. Further details about the MCIR scheme can be found in Ramesh and Deshpande (2007).
The \( \psi \) moment of equation (5) will lead to the Modified Kinetic split flux Euler equation

\[
\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}(G_X^+ + M) + \frac{\partial}{\partial y}(G_Y^+ + M) + \frac{\partial}{\partial y}(G_Y^- - M) = 0,
\]

where \( G_X^+ \) and \( G_Y^+ \) are modified KFVS split fluxes. These can be represented in terms of the usual KFVS fluxes as

\[
G_X^+ = \frac{1}{2}\{G_X \pm \phi_1(G_X^+ - G_X^-)\},
\]

\[
G_Y^+ = \frac{1}{2}\{G_Y \pm \phi_2(G_Y^+ - G_Y^-)\}.
\]

### 2.2. Boundary condition

The boundary conditions (wall or far-field) are implemented in a local Cartesian coordinate system. The treatment of the boundary condition is developed at a molecular level and by taking suitable moments; we obtain the boundary condition at the macroscopic (Euler) level. This procedure is called the ‘kinetic treatment of the boundary condition’. To implement a wall boundary condition (surface tangency condition), the Maxwellian velocity distribution function \( F \), at any point ‘i’ lying on the outer boundary surface (Figure 2(b)), is constructed as the union of two half Maxwellians, corresponding to the incoming and outgoing particles. The incoming particles have a negative velocity with respect to the direction normal to the boundary. The velocity distribution function \( F_{\text{in}} \) for this part corresponds to the free stream conditions. The outgoing particles have a positive velocity in the normal direction. For the outgoing particles the velocity distribution function \( F_{\text{out}} \) is constructed from the interior region.

\[
F_{\text{in}} = F_{\text{in}} \cup F_{\text{out}}.
\]

Details about the boundary conditions can be found in the work of Mandal and Deshpande (1994) and Ramesh and Deshpande (2001).
where \( p_1, p_2, q_1 \) and \( q_2 \) are geometrical constants (see Appendix A) and \( \Delta(.)_k = (.)_k - (.)_j \). The superscript ‘+’ and ‘−’ indicate that split stencils are used to evaluate corresponding constants (see Figures 1(a–d)). For simplicity, we take the first term of the spatial derivative of equation (7),

\[
\frac{\partial (GX^+)^{n+1}}{\partial X} = p_1^+ \sum_k (GX^+_k - GX^+_i)^{n+1} \Delta x_k \\
+ p_2^+ \sum_k (GX^+_k - GX^+_i)^{n+1} \Delta y_k. \tag{10}
\]

After temporal linearization for the current time level \( n \),

\[
(GX^\pm)^{n+1} = (GX^\pm)^n + A^\pm \cdot \Delta t \cdot U \tag{11}
\]

where \( A^\pm \) is split flux Jacobian for the \( x \) direction and \( \Delta t \cdot U = (U^{n+1} - U^n) \). From equation (10) and equation (11), we have

\[
\frac{\partial (GX^+)^{n+1}}{\partial X} = \begin{pmatrix}
p_1^+ \sum_k (A^+_k \cdot \Delta t \cdot U_k - A^+_i \cdot \Delta t \cdot U_i) \cdot \Delta x_k \\
+ p_2^+ \sum_k (A^+_k \cdot \Delta t \cdot U_k - A^+_i \cdot \Delta t \cdot U_i) \cdot \Delta y_k
\end{pmatrix}
+ \begin{pmatrix}
p_1^- \sum_k \Delta G^+_k \cdot \Delta x_k \\
+ p_2^- \sum_k \Delta G^+_k \cdot \Delta y_k
\end{pmatrix}
\tag{12}
\]

The terms in the first set of square brackets of equation (12) are termed as the implicit operator \( \text{Im}(R) \), while the terms in the second set of square brackets are termed as the explicit operator \( R^n \) (or the explicit residual). The implicit operator requires computation of split flux Jacobians. We make use of a matrix-free approach (Luo et al., 1998), which is very economical as compared to the expensive computation of split flux Jacobians. In this approach, the product of split flux Jacobians (or flux Jacobians) with incremental conserved variable vector is approximated by an incremental change in the split flux vector in time:

\[
A^\pm \cdot \Delta t \cdot U_k = \frac{\partial (GX^\pm)}{\partial U_k} \cdot \Delta t \cdot U_k = GX^\pm_k (U + \Delta U) - GX^\pm_k U
= \Delta G^\pm_k 
\]

With this, the implicit operator term of equation (12) becomes

\[
\text{Im}(R) = p_1^+ \sum_k \Delta t \cdot G^+_k \cdot \Delta x_k + p_2^+ \sum_k \Delta t \cdot G^+_k \cdot \Delta y_k
- A^+_i \cdot \Delta t \cdot U_i \left( p_1^+ \sum_k \Delta x_k + p_2^+ \sum_k \Delta y_k \right) \tag{13}
\]

The terms in the first set of square brackets of equation (13) correspond to the lower and upper matrix of the
implicit operator, which is clubbed along with the explicit operator. The terms in the second set of square brackets correspond to the diagonal term of the implicit operator. Likewise, working with the other spatial derivative terms of split fluxes from equation (7), we arrive at a discretized equation corresponding to equation (7) as

\[
\Delta_i U_i = - R_i^a - (E + C), \tag{14}
\]

where, \( A_i^\pm \) and \( B_i^\pm \) are split flux Jacobians in the \( x \) and \( y \) directions, respectively. \( R_i^a \) is the explicit residual (or explicit operator) and \( E + C \) are terms which correspond to the lower and upper matrix of the implicit operator (see Appendix B). Following Jameson and Yoon’s split flux Jacobians approximation based on the spectral radius,

\[
A_i^\pm = \frac{A_i \pm \rho_A I}{2}, \quad B_i^\pm = \frac{B_i \pm \rho_B I}{2},
\]

where \( \rho_A \) and \( \rho_B \) are the spectral radius in the \( x \) and \( y \) directions, respectively.

The left side of equation (14) is given by

\[
D_i, \Delta_i U_i = \left[ \begin{array}{c} \frac{\partial}{\partial x} \Delta_i U_i \\ \frac{\partial}{\partial y} \Delta_i U_i \\ - \frac{\partial}{\partial x} \Delta_i U_i \\ - \frac{\partial}{\partial y} \Delta_i U_i \end{array} \right] = \Delta_i U_i
\]

\[
= \left[ \begin{array}{c} \frac{I}{\Delta} \sum_{k} p_i^+ \Delta x_k + p_i^- \sum_{k} \Delta y_k \\ \sum_{k} q_i^+ \Delta x_k + q_i^- \sum_{k} \Delta y_k \\ \sum_{k} p_i^+ \Delta x_k - p_i^- \sum_{k} \Delta y_k \\ - \sum_{k} q_i^+ \Delta x_k - q_i^- \sum_{k} \Delta y_k \end{array} \right]
\]

\[
\Delta U^a = D_{ii}^{-1} \left[ \begin{array}{c} - R_i^a - (E + C) + S \end{array} \right]; \tag{17a}
\]

\[
\Delta U^{n+1} = D_{ii}^{-1} \left[ \begin{array}{c} - R_i^a - (E + C) - (E + C) + S \end{array} \right]. \tag{17b}
\]
It is to be noted that $D_{ii}^{-1} = 1/D_{ii}$. The spectral radius is taken as $\rho_A = |u| + c$, $\rho_B = |v| + c$, where $c$ is the speed of sound and $u$ and $v$ are the fluid velocities in the $x$ and $y$ directions, respectively (for more details, see Singh, 2010; Singh, Ramesh, & Balakrishnan, 2010, 2012).

4. Results and discussion

4.1. 2D test cases

To test the implicit and explicit solver on the NACA 0012 airfoil, a point cloud consisting of 9600 points is created (see Figure 3). There are a total of 160 points on the surface of the airfoil. The far-field distance is taken as 10 times the chord length. All the computations are carried out on Silicon Graphics International (SGI) Altix ICE clusters with a clock speed of 2.93 GHz. The solution is assumed to have converged after 5 decade density residue fall. The Courant–Friedrichs–Lewy (CFL) number for explicit schemes is limited by stability considerations. In the present explicit scheme, the CFL number is taken as 0.8. A larger CFL number can be used in the case of implicit schemes. Figure 4 shows the effect of the CFL number on the convergence of the present implicit scheme for flow over a NACA 0012 airfoil at Mach 0.65 with an angle of attack of 1.86°. It is observed that the convergence speed-up increases with the CFL number up to 1000. Increasing the CFL number beyond 1000 does not have any effect on the convergence. In the present implicit scheme, the CFL number is kept at 1000.

4.1.1. Subsonic case

The subsonic test case is the flow over the NACA 0012 airfoil at Mach 0.65 with an angle of attack of 1.86°. Wind tunnel experimental data is available in Harris (1981). The computed surface pressure coefficient (Cp) distribution along the chord line is compared with the wind tunnel data in Figure 5(a). It is evident that explicit and implicit meshless computed data are in good agreement with the wind tunnel data. The iso-pressure contour plots are smooth and identical for the explicit and implicit schemes (Figure 5(b)). The convergence history for the density residue and lift coefficient are given in Figures 6(a and b), respectively. It is found that the present implicit scheme takes about 80 seconds to converge, as opposed to 1092 seconds in the RK4 explicit scheme. So, in this case, the implicit scheme is 13.6 times faster than the RK4 explicit scheme. Table 1 summarizes this test case.
Figure 5. A NACA 0012 airfoil at $M_\infty = 0.65, \alpha = 1.86^\circ$: (a) the surface pressure coefficient distribution along the chord line and (b) the normalized pressure contour.

Figure 6. A NACA 0012 airfoil at $M_\infty = 0.65, \alpha = 1.86^\circ$: the convergence history for (a) density residue vs time and (b) lift coefficient (CL) vs time.

Table 1. Subsonic test case ($M_\infty = 0.65, \alpha = 1.86^\circ$).

|                | Explicit | Implicit | Exp. (Harris, 1981) |
|----------------|----------|----------|---------------------|
| Iterations     | 13,920   | 1911     |                     |
| Time (seconds) | 1092.5   | 80.1     |                     |
| Lift coefficient | .253    | .255    | .233               |
| Speed-up       | 13.6     |          |                     |

4.1.2. Transonic case

The standard AGARD test case (Viviand, 1985) of transonic flow past a NACA 0012 airfoil at Mach 0.85 with an angle of attack of $1^\circ$ was chosen for demonstrating the present methodology under the transonic regime. In this test case, shocks appear on both the lower and upper surfaces (Figure 7(a)), which are captured by the present methodology reasonably well. The pressure contour plot is shown in Figure 7(b). The density convergence history is plotted in Figures 8(a and b) against the time and number of iterations, respectively. A speed-up of 7.2 is obtained by using the implicit procedure as compared to the RK4 explicit procedure (see Table 2). Table 2 also compares the lift and drag coefficients with the AGARD results. The explicit and implicit meshless predictions are well within the range of the AGARD data.
M.K. Singh et al.

Figure 7. A NACA 0012 airfoil at $M_\infty = 0.85, \alpha = 1.0^\circ$: (a) the surface pressure coefficient distribution along the chord line and (b) the normalized pressure contour.

Figure 8. A NACA 0012 airfoil at $M_\infty = 0.85, \alpha = 1.0^\circ$: the convergence history for (a) density residue vs time and (b) density residue vs iterations.

Table 2. Transonic test case ($M_\infty = 0.85, \alpha = 1.0^\circ$).

|                  | Explicit | Implicit | AGARD AR-211 |
|------------------|----------|----------|---------------|
| Iterations       | 13,540   | 2824     |               |
| Time (seconds)   | 885.0    | 123.2    |               |
| Lift coefficient | .368     | .361     | .330–.389     |
| Drag coefficient | .0515    | .0513    | .464–.0590    |
| Speed-up         | 7.2      |          |               |

4.1.3. Supersonic case

The test case here is the flow past a NACA 0012 airfoil at Mach 1.2 and with an angle of attack of $7^\circ$. In Figure 9(a), the computed surface pressure coefficient distribution along the chord line shows good agreement with Schmidt and Jameson’s computations from the AGARD report (Viviand, 1985). Figure 9(b) shows the pressure contour in the flow field. The bow shock ahead of the leading edge is captured reasonably well. The convergence of the lift and drag coefficients is shown in Figures 10(a and b), respectively. Table 3 compares the lift and drag coefficients with the AGARD results.

4.2. Convergence speed vs memory

One of the key attractions of implicit solvers is the convergence speed-up, which varies according to the Mach number. The convergence speed-up obtained for subsonic
Figure 9. A NACA 0012 airfoil at $M_\infty = 1.2$, $\alpha = 7.0^\circ$: (a) the surface pressure coefficient distribution along the chord line and (b) the normalized pressure contour.

Figure 10. A NACA 0012 airfoil at $M_\infty = 1.2$, $\alpha = 7.0^\circ$: the convergence history for (a) the lift coefficient (CL) vs time and (b) the drag coefficient (CD) vs time.

Table 3. Supersonic test case ($M_\infty = 1.20$, $\alpha = 7.0^\circ$).

|               | Explicit | Implicit | AGARD AR-211 |
|---------------|----------|----------|---------------|
| Lift coefficient | .5399    | .5342    | .5138--.5280  |
| Drag coefficient | .1586    | .1581    | .1531--.1543  |

flow is close to 14, as compared to 7 for transonic flow. However, the price paid for this convergence acceleration is an increase in memory requirement. Implicit solvers need to store more parameters than with the explicit counterpart. It is observed that for all 2D test cases mentioned above, the implicit solver requires approximately 80% more memory than the explicit solver.

4.3. 3D test case

The performance of the implicit 3D solvers is tested for transonic flow on an ONERA M6 wing. At Mach 0.84 and an angle of attack of 3.06°, lambda shock occurs on the upper surface of the ONERA wing, and this is the standard test case for 3D CFD codes. The experimental results are available in Schmitt and Charpin (1979). The clipped delta wing with deflected aileron is selected to
demonstrate the capability of a meshless solver to handle the small gaps effectively. The experimental data is taken from Hess, Cazier, and Wyne (1986). All of these computations are carried out on Centre for Mathematical Modeling and Computer Simulations (CMMACS) SGI clusters (Altix system).

4.3.1. ONERA M6 Wing

The ONERA M6 wing is a swept, semi-span wing with no twist. It uses a symmetric airfoil using the ONERA D section. The ONERA M6 wing is a standard CFD validation case because of its simple geometry combined with a complex transonic flow. A large number of CFD codes have been validated against this test case over the years. Euler simulation of flow over ONERA M6 wing is performed using explicit and implicit meshless solvers. To obtain the point distribution, an unstructured grid is created in the flow domain and only point coordinate information is retained for the meshless solver. Details about flow and point distribution are given in Table 4. The point distribution around the wing is shown in Figure 11. For a higher order of accuracy, the MCIR factor $p$-value should be as close to unity as possible ($p < 1$). For the ONERA M6 wing, a maximum $p$-value of only .56 is achieved using an explicit scheme, as compared to .63 for an implicit scheme. Each iteration of the RK4 explicit scheme is about 1.6 times costlier than the implicit scheme. It is found that the implicit scheme converges in 1000 iterations against 4500 iterations in the explicit scheme. So, the speed-up obtained is more than 7 in this case. However, only the implicit results are presented here because of the better accuracy achieved through the higher $p$-value.

The comparison of the pressure coefficient (Cp) distribution along a local chord line (in a fraction of local chord length, $X/C$) between computations and experiments are shown in Figures 12 and 13 for different span wise locations ranging from 20% span to 95% span. Figures 12(a and b) show Cp comparison plots at the locations of 20% and 44% span, respectively. It is observed that implicit computation is able to accurately capture both the pressure peak and the shock location quite satisfactorily.

Figures 13(a–c) show $C_p$ comparison plots at the locations of 65%, 80% and 90% span, respectively. It is observed that implicit computation can capture the pressure peak and second shock location very accurately. At

| Parameter                | Value |
|-------------------------|-------|
| Mach number ($M_∞$)     | 0.84  |
| Angle of attack ($α$)   | 3.06° |
| Total number of points  | 632,164 |
| Points on wing surface  | 104,618 |
| Far-field along span     | 5 (Span length) |
| Far-field upstream and downstream | 10 (Root chord length) |

Figure 11. Point distribution around an ONERA M6 wing.

(a) (b)

Figure 12. The surface pressure coefficient distribution along the local chord line at (a) 20% and (b) 44% span on an ONERA M6 wing at $M_∞ = 0.84, α = 3.06°$. 
the 90% span location, there is a single shock which is captured very well by computation. Figure 13(d) shows a Cp comparison plot at the location of 95% span. This section also has a single shock similar to the 90% span location. The computed results predicted the shock location and pressure peaks very well. Finally, a normalized pressure (with respect to free stream pressure) contour plot is given in Figure 14. The lambda-shape shock can be noticed very clearly in this figure. Present computation shows that LSKUM with MCIR splitting can capture all the flow features very well for the case of transonic flow past an ONERA M6 wing.

4.3.2. **Clipped delta wing with aileron deflection of 6°**

Figure 15(a) shows the plan-form of the cropped delta wing. The leading edge backward sweep is 50.4°. The aileron is hinged from 56.5% semi-span to 82.9% of semi-span and at 80% of the local chord from the leading edge. The aileron deflection is 6°. The slot gap between the deflected aileron and the main wing is modeled as it is. The present computation uses 601,359 points (for details of the flow and point distribution, see Table 5). The information about five sectional locations along the span, where experimental Cp data is available, is given in Figure 15(b) and Table 6. At location D, which lies in the middle of the deflected aileron, present Euler computations are also compared with the Euler and Reynolds-Averaged Navier-Stokes (RANS) of Li, Zhu, Chen, and Li (1999) using finite volume code. The finite volume code of Li et al.
Table 5. Flow and point distribution parameter for a clipped delta wing.

| Parameter                  | Value          |
|----------------------------|----------------|
| Mach number ($M_\infty$)   | 0.90           |
| Angle of attack ($\alpha$) | $1^\circ$, $3^\circ$ |
| Total number of points     | 601,359        |
| Points on wing surface     | 143,413        |
| Far-field along span       | 5 (Span length) |
| Far-field upstream and downstream | 10 (Root chord length) |

Table 6. Span-wise section locations.

| Location | % of span | Comments                           |
|----------|-----------|------------------------------------|
| A        | 33.2      | Section on main wing               |
| B        | 54.1      | On the main wing but close to the deflected aileron |
| C        | 58.7      | On deflected aileron               |
| D        | 69.4      | Middle of the deflected aileron    |
| E        | 85.1      | On main wing but close to the deflected aileron |

(1999) uses a Van Leer flux vector splitting scheme for the discretization of in-viscid fluxes. This code uses 120,000 and 506,250 grid cells for the Euler and NS computations, respectively.

The $C_p$ comparisons between the present meshless Euler computations, the wind tunnel experiments (Hess et al., 1986) and the earlier Euler and RANS computations (Li et al., 1999) are presented in Figures 16(a and b) for angles of attack of $1^\circ$ and $3^\circ$, respectively at section D (see Figure 15(b)). Both of the Euler computations over-predict the suction pressure at the deflection location. However, the RANS computation of Li et al. (1999), which is closer...
to the experiments, does not show pressure suction to this extent at this location. The present meshless Euler computations capture the leading edge suction peak quite well.

The \( C_p \) comparison between the implicit computations and experiments, for locations A, B, C and E are presented in Figures 17a, 18a, 19a and 20a, respectively for an angle of attack of 1°, and in Figures 17b, 18b, 19b and 20b for an angle of attack of 3°. The computed \( C_p \) matches reasonably well with the experiments except at location C (Figures 19a and b), which is on the deflected portion of the aileron (see Figure 15(b)). The present Euler computations predict an increase in suction pressure at the deflected location for both angles of attack. This trend is similar to...
what has been observed at location D. For all locations, there is not enough experimental data available between 80 and 90% of the local chord length. The leading edge suction pressure is more for $\alpha = 3^\circ$, in comparison with $\alpha = 1^\circ$ at all sections. Normalized pressure (with respect to free stream pressure) contour plots are given in Figures 21(a and b) for $\alpha = 1^\circ$ and $\alpha = 3^\circ$, respectively. On the upper surface of the wing, a relatively lower pressure region is observed in a larger area at $\alpha = 3^\circ$.

5. Conclusion
An implicit scheme for a meshless solver based on the kinetic upwind method is presented. Advances made in the implicit finite volume methodology, like the matrix-free approach and Jameson and Yoon’s split flux Jacobian for the diagonal-dominant implicit procedure are adopted for the meshless framework. The capability of the implicit scheme is demonstrated by the computing flow past a NACA 0012 airfoil under subsonic, transonic and supersonic regimes. A convergence acceleration up to 14 is obtained over the RK4 explicit scheme. The results obtained are in good agreement with the wind tunnel experimental data and other reliable CFD computations. However, the memory requirement for the implicit solver is approximately 80% more than that of the RK4 explicit solver. Further, the accuracy and robustness of the scheme in 3D is demonstrated by computing the flow past an ONERA M6 wing and a clipped delta wing with an aileron deflection of 6°. The computed results show good agreement with the wind tunnel experimental data and other CFD computations.

At present, this study is limited to a meshless in-viscid Euler solver. We plan to implement this methodology in
a viscous meshless solver as well. Reducing the memory requirement of the implicit solver is also one of the priorities.

Acknowledgment

The authors would like to thank Mr Richard A. Evans, College of Engineering, Swansea University, Swansea, United Kingdom for editing the manuscript and suggesting corrections and improvements.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

Anandhanarayanan, K. (2003). Development and application of a Gridfree kinetic upwind solver to multibody configurations (Ph.D. Thesis), Dept. of Aerospace Engg., Indian Institute of Science, Bangalore, India.

Anandhanarayanan, K., Nagarathinam, M., & Deshpande, S. M. (2004). An entropy variable based grid-free solver with LU-SGS accelerator. 24th Intl. Congress of Aeronautical Sciences, Yokohama, Japan. ICAS2004-2.4.3.

Anil, N. (2008). Optimal control of numerical dissipation in modified KFVS (m-KFVS) using Discrete Adjoint Method (Ph.D. Thesis). Dept. of Aerospace Engg., Indian Institute of Science, Bangalore, India.

Balakrishnan, N., & Sridar, D. (2001). On a generalised finite difference framework and its order of accuracy. Computational Fluid Dynamics 2000, Springer New York: 797–798.

Batina, J. T. (1993). A gridless Euler/Navier–Stokes solution algorithm for complex aircraft applications. AIAA Paper 1993-0333.

Chiu, E. K., Wang, Q., & Jameson, A. (2011). A conservative meshless scheme: General order formulation and application to euler equations. AIAA Paper 2011-0651.

Demkowicz, L., Karafiat, A., & Liszka, T. (1984). On some convergence results for FDM with irregular mesh. Computer Methods in Applied Mechanics and Engineering, 42, 343–355.

Deshpande, S. M. (1986a). On the Maxwellian distribution, symmetric form and entropy conservation for the euler equations. NASA TP 2583.

Deshpande, S. M. (1986b). A second order accurate kinetic theory based method for inviscid compressible flow. NASA TP 2613.

Deshpande, S. M., Anandhanarayanan, K., Praveen, C., & Ramesh, V. (2002). Theory and application of 3-D LSKUM based on entropy variables. International Journal for Numerical Methods in Fluids, 40, 47–62.

Deshpande, S. M., Kulkarni, P. S., & Ghosh, A. K. (1998). New developments in kinetic schemes. Journal of Computers and Mathematics with Applications, 35(1/2), 75–93.

Dominic, D. J. C. (2004). A one point shock capturing kinetic scheme for hyperbolic conservation laws (M.E. Thesis). Dept. of Aerospace Engg., IISc, Bangalore, India.

Ghosh, A. K., & Deshpande, S. M. (1995). Least Squares Kinetic Upwind Method for inviscid compressible flows. AIAA Paper 1995–1735.

Gingold, R. A., & Monaghan, J. J. (1977). Smoothed particle hydrodynamics - Theory and applications to non-spherical stars. Monthly Notices of the Royal Astronomical Society, 181, 375–389.

Harris, C. D. (1981). Two-dimensional aerodynamic characteristics of the NACA 0012 Airfoil in the Langley 7-foot transonic pressure tunnel. NASA Technical Memorandum 81927.

Hess, R. W., Cazier, F. W., & Wyne, E. C. (1986). Steady and unsteady transonic pressure measurements on a clipped delta wing for pitching and control-surface oscillations. NASA TP 2594.

Jameson, A., & Yoon, S. (1987). Lower-upper implicit schemes with multiple grids for the euler equations. AIAA Journal, 25(7), 929–935.

Katz, A., & Jameson, A. (2009a). A comparison of various meshless schemes within a unified algorithm. AIAA paper 2009-596.

Katz, A., & Jameson, A. (2009b). A meshless volume scheme. AIAA Paper 2009–3544.

Katz, A., & Jameson, A. (2009c). Multicloud: multigrid convergence with a meshless operator. Journal of Computational Physics, 228, 5237–5250.

Li, J., Zhu, Z. Q., Chen, Z. M., & Li, H. M. (1999). The Euler and Navier-Stokes solutions of a 3-D wing with aileron. Acta Mechanica, 138, 51–59.

Liszka, T., & Orkisz, J. (1980). The finite difference method at arbitrary irregular grids and its application in applied mechanics. Computers and Structures, 11(1/2), 83–95.

Lucy, L. B. (1977). A numerical approach to the testing of fission hypothesis. The Astronomical Journal, 82, 1013–1024.

Luo, H., Baum, J. D., & Lohner, R. (1998). A fast, matrix-free implicit method for compressible flows on unstructured grid. Journal of Computational Physics, 146, 664–690.

Mahindra, A. K., Singh, R. K., & Gouthaman, G. (2011). Meshless kinetic upwind method for compressible, viscous rotating flows. Computers and Fluids, 46(1), 325–332.

Mandal, J. C., & Deshpande, S. M. (1994). Kinetic flux vector splitting method for euler equations. Computers and Fluids, 23(2), 447–478.

Monaco, A. D., Manenti, S., Gallati, M., Sibilla, S., Agate, G., & Guandalini, R. (2011). SPH modeling of solid boundaries through a semi-analytic approach. Engineering Applications of Computational Fluid Mechanics, 5(1), 1–15.

Munikrishna, N. (2007). On viscous flux discretization procedures for finite volume and meshless solvers. Ph. D. Thesis, Dept. of Aerospace Engg., Indian Institute of Science, Bangalore, India.

Munikrishna, N., & Balakrishnan, N. (2007). Turbulent Flow Computations using Meshless Solver LSFD-U. Proceedings of Seventh Asian Computational Fluid Dynamics Conference (ACFD7) Bangalore.

Ninawe, A., Munikrishna, N., & Balakrishnan, N. (2006). Viscous Flow Computations Using a Meshless Solver, LSFD–U. Computational Fluid Dynamics 2004, Springer–Verlag, Berlin: 509–514.

Pu, J. H., Shao, S., Huang, Y., & Hussain, K. (2013). Evaluations of SWEs and SPH numerical modelling techniques for dam break flows. Engineering Applications of Computational Fluid Mechanics, 7(4), 544–563.

Ramesh, V. (2001). Least Squares Grid Free Kinetic Upwind Method (Ph.D. Thesis). Dept. of Aerospace Engg., Indian Institute of Science, Bangalore, India.

Ramesh, V., & Deshpande, S. M. (2001). Least squares kinetic upwind method on moving grids for unsteady euler computations. Computers and Fluids, 30, 621–641.

Ramesh, V., & Deshpande, S. M. (2004). Low dissipation grid free upwind kinetic scheme with modified CIR splitting. Fluid Mechanics report, FM-20, Dept of Aerospace Engg., Indian Institute of Science, Bangalore, India.
Appendix A

In least square approximation, the flux derivative for point ‘i’ can be written as

\[
\begin{align*}
\frac{\partial G_X}{\partial x} &= p_1 \sum_k \Delta G_X \Delta x_k + p_2 \sum_k \Delta G_X \Delta y_k \\
\frac{\partial G_Y}{\partial x} &= q_1 \sum_k \Delta G_Y \Delta x_k + q_2 \sum_k \Delta G_Y \Delta y_k 
\end{align*}
\]

where ‘k’ is point in connectivity, and \(p_1, p_2, q_1\) and \(q_2\) are the geometrical constants given by

\[
p_1 = \left( \sum_k \Delta x_k \Delta y_k \right)^2 - \sum_k \Delta x_k^2 \sum_k \Delta y_k^2,
\]

\[
p_2 = \frac{\sum_k \Delta x_k \Delta y_k}{\left( \sum_k \Delta x_k \Delta y_k \right)^2 - \sum_k \Delta x_k^2 \sum_k \Delta y_k^2},
\]

\[
q_1 = \frac{\left( \sum_k \Delta x_k \Delta y_k \right)^2 - \sum_k \Delta x_k^2 \sum_k \Delta y_k^2}{\sum_k \Delta y_k^2},
\]

\[
q_2 = \frac{\sum_k \Delta x_k \Delta y_k}{\left( \sum_k \Delta x_k \Delta y_k \right)^2 - \sum_k \Delta x_k^2 \sum_k \Delta y_k^2}.
\]

Appendix B

The explicit operator is given by

\[
R^n = \left[ \begin{array}{c}
p_1^+ \sum_k \Delta G_X^+ \Delta x_k + p_2^+ \sum_k \Delta G_X^+ \Delta y_k \\
+ q_1^+ \sum_k \Delta G_Y^+ \Delta x_k + q_2^+ \sum_k \Delta G_Y^+ \Delta y_k \\
+ q_1^- \sum_k \Delta G_Y^- \Delta x_k + q_2^- \sum_k \Delta G_Y^- \Delta y_k \\
\end{array} \right]
\]

The lower and upper implicit operator is given by

\[
E + C = \left[ \begin{array}{c}
p_1^+ \sum_k \Delta G_X^+ \Delta x_k + p_2^+ \sum_k \Delta G_X^+ \Delta y_k \\
+ q_1^+ \sum_k \Delta G_Y^+ \Delta x_k + q_2^+ \sum_k \Delta G_Y^+ \Delta y_k \\
+ q_1^- \sum_k \Delta G_Y^- \Delta x_k + q_2^- \sum_k \Delta G_Y^- \Delta y_k \\
\end{array} \right]
\]

The scalar diagonal matrix is given by

\[
D_{ii} = \left[ \begin{array}{c}
\frac{1}{\Delta x_i} - \frac{p_2}{2} \\
- \frac{p_1}{\Delta x_i} - p_2 \sum_k \Delta y_k \\
q_1^+ \sum_k \Delta x_k + q_2^+ \sum_k \Delta y_k \\
- q_1^- \sum_k \Delta x_k - q_2^- \sum_k \Delta y_k \\
\end{array} \right]
\]

The term ‘S’ in equation (16) is given by

\[
S = \left[ \begin{array}{c}
\frac{\Delta G_X}{2} \left( p_1^+ \sum_k \Delta x_k + p_2^+ \sum_k \Delta y_k \\
+ q_1^+ \sum_k \Delta x_k + q_2^+ \sum_k \Delta y_k \\
+ q_1^- \sum_k \Delta x_k + q_2^- \sum_k \Delta y_k \\
\end{array} \right)
\]

In the above expressions, \(\Delta(.)_k = (.)_k - (.)_i\) and \(\Delta_i(.)_k = (.)_k^{i+1} - (.)_k^i\).