Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Journal of Computational Physics 189 (2003) 1–29

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

An upwind finite difference scheme for meshless solvers

D. Sridar, N. Balakrishnan *

CFD Centre, Department of Aerospace Engineering, Indian Institute of Science, Bangalore 560 012, India

Received 8 March 2002; received in revised form 18 December 2002; accepted 24 February 2003

Abstract

In this paper, we present a new upwind finite difference scheme for meshless solvers. This new scheme, capable of working on any type of grid (structure, unstructured or even a random distribution of points) produces superior results. A means to construct schemes of specified order of accuracy is discussed. Numerical computations for different types of flow over a wide range of Mach numbers are presented. Also, these results were compared with those obtained using a cell vertex finite volume code on the same grids and with theoretical values wherever possible. The present framework has the flexibility to choose between various upwind flux formulas.

Keywords: Meshless method; Finite difference; Upwind scheme; Least squares

1. Introduction

One of the major difficulties in the computation of flows involving complex real life configuration is grid generation. In the past 10 years remarkable progress has been made both in terms of grid generation techniques and flow solvers which would alleviate this problem. The successful use of adaptive unstructured and Cartesian mesh calculations can be cited as examples in this regard. Though the computational effort involved in such calculations is considerably more compared to the routine structured mesh calculations, the ever increasing speed of the computers has made these computations more realistic. One other class of methods called the 'Grid-free methods' also address the question of computation of flow past complex configurations. All these methods require, to approximate derivatives at any given node, is the information at a cloud of grid points around that node. This set of grid points can be generated using any means; structured, unstructured or cartesian mesh generator and hence the name "Grid-free method."

In the past two decades considerable amount of research has been carried out in this area, popularly referred to as "Gridless CFD" [17]. One of the earliest work pertains to the paper on "Generalized finite difference method" by Chung [10]. The fundamental idea explored in this work related to the use of Taylor series expansion for obtaining the discrete approximation to the derivatives at any given

* Corresponding author.
E-mail addresses: sridar@aero.iisc.ernet.in (D. Sridar), nbalak@aero.iisc.ernet.in (N. Balakrishnan).

0021-9991/03/$ - see front matter © 2003 Elsevier Science B.V. All rights reserved.
doi:10.1016/S0021-9991(03)00197-9

point has not undergone any change. On the other hand at the implementation level significant modifications have been experimented with, leading to the success/failure of the aforesaid procedure. As example we can cite the work of Deshpande et al. [11] and Batina [8]. Both employ a least-squares procedure for solving the resulting over determined system of equations in contrast to the work of Chung [10], where a stencil of grid points just adequate to solve for the derivatives appearing in the truncated Taylor series is employed. While the work of Deshpande et al. [11] deals with an upwind implementation based on Kinetics Theory of Gases, resulting in least-squares kinetic upwind method (LSKUM) [12], that of Batina [8] is based on a centered scheme using artificial dissipation. In this paper, we present a new Least-Squares-based Upwind Finite Difference method [2,5,6], referred to as LSFD-U. The new method unlike the LSKUM which makes use of one-sided upwind stencil of grid points uses a global stencil of grid points. The fact that the present upwind method employs a global stencil of grid points can be considered as one of the most important advantages in comparison to LSKUM. Also, the present method has the flexibility to choose between different flux formulas like Roe [26], van Leer [29], KFVS [18], AUSM [19], etc. The other interesting development in this area is due to Morinishi [20,21] and Löhner et. al [25]. They have used "weighted least-squares" approach to find the derivatives at a node. Though the use of a *mid-point* between a given node and its neighbour to determine an upwind flux is similar to the strategy presented in this work, the effective upwind direction is arbitrary. In present work, the upwinding is done along each ray joining the node and its neighbour. In [21] it is remarked that it is difficult to prove the order of accuracy of such schemes. On the contrary in this paper we present a means to construct schemes of specified order of accuracy. Based on the development in LSFD-U, a new kinetic theory-based scheme employing a global stencil of grid points has also been developed. An interested reader is referred to [24]. Other methods outside the realm of finite difference method, for fluid flow computations, are reproducing kernel particle method (RKPM) due to Liu et al. [31] and meshless local Petrov–Galerkin (MLPG) due to Atluri [16]. The RKPM is also a meshless particle (Lagrangian) method like cubic interpolation with volume/area co-ordinates (CIVA) due to Tanaka [23]. The CIVA uses cubic interpolation pseudo-particle (CIP) algorithm to obtain high accuracy.

In Section 2 a general introduction to a least-squares-based update procedure is presented, followed by a discussion on its upwind implementation in Section 3. The LSFD-U procedure is introduced in Section 4. Its implementation in 2D, order of accuracy and its variants are presented in Section 5. The results obtained using LSFD-U procedure and its variants for a wide range of Mach numbers are discussed in Section 6, as well as the comparison with results obtained using cell vertex finite volume procedure.

2. Least-squares-based update procedure for fluid flow problems

Consider the 2D Euler equation of gas dynamics

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

where $\mathbf{U}$ is the vector of conserved variables given by $\mathbf{U} = [\rho \quad \rho u \quad \rho v \quad e]^{\mathrm{T}}$, $\mathbf{f}$ and $\mathbf{g}$ represent the fluxes in the x and y co-ordinate directions, respectively. The fluxes $\mathbf{f}$ and $\mathbf{g}$ are given by $\mathbf{f} = [\rho u \quad \rho u^2 + p \quad \rho uv \quad (e+p)u]^{\mathrm{T}}$ and $\mathbf{g} = [\rho v \quad \rho uv \quad \rho v^2 + p \quad (e+p)v]^{\mathrm{T}}$. [1] In the above expressions ρ, u, v and p stand for density, x and y components of the fluid velocity and pressure, respectively. They are related to the energy per unit volume through the relation $e = p/(\gamma - 1) + (\rho(u^2 + v^2)/2)$. We seek to solve

[1] Throughout this paper, we use $\mathbf{f}$ and $\mathbf{g}$ to represent the flux vectors and f and g their components, respectively.

the Euler equations numerically using the method of least squares. In this section we introduce the least-squares procedure and in the subsequent sections discuss the upwind implementation of the least-squares procedure.

The least-squares finite difference update formula operates on a cluster of grid points. A typical 2D cluster of grid points is shown in Fig. 1. The state update at the point o using a suitable time integration procedure, requires the values of $\mathbf{f}_x$ and $\mathbf{g}_y$ at that point. We demonstrate the least-squares procedure employed in approximating the flux derivatives at o by introducing an arbitrary function $\phi(x,y)$, the discrete values of which are available at the nodes. The function value in the neighborhood of o (say at node i) can be estimated using a truncated Taylor series

$$\phi_i = \phi_o + \sum_{q=1}^{l} \sum_{m=0}^{q} \binom{q}{m} \frac{\Delta x_i^{q-m} \Delta y_i^m}{q!} \frac{\partial^q \phi}{\partial x^{q-m} \partial y^m}, \tag{2}$$

with $\Delta(\cdot)_i = (\cdot)_i - (\cdot)_0$. Let n represent the number of neighbors of o. Eq. (2), for $n > l(l+3)/2$, represents an over-determined system of equations, the solution of which can be determined using the method of least squares. The aforesaid system of equations can be represented in matrix form as

$$\mathbf{Ad} = \mathbf{b}. \tag{3}$$

While $\mathbf{d}$ and $\mathbf{b}$ represent vectors of dimension $l(l+3)/2$ and n, respectively, $\mathbf{A}$ represents $(n \times (l(l+3)/2))$ matrix. The elements of $\mathbf{A}$ and $\mathbf{d}$ can be built from the following vectors, defined for a given q,

$$\begin{aligned} a_{iq} &= \begin{bmatrix} \frac{\Delta x_i^q}{q!} & \frac{\Delta x_i^{q-1}}{q!} \Delta y_i & \dots & \frac{\Delta y_i^q}{q!} \end{bmatrix}, \\ d_q &= \begin{bmatrix} \frac{\partial^q \phi}{\partial x^q} & q \frac{\partial^q \phi}{\partial x^{q-1} \partial y} & \frac{q(q-1)}{2} \frac{\partial^q \phi}{\partial x^{q-2} \partial y^2} & \dots & \frac{\partial^q \phi}{\partial y^q} \end{bmatrix}_o^{\mathrm{T}}. \end{aligned} \tag{4}$$

The ith component of vector $\mathbf{b}$ is given by $b_i = \Delta\phi_i$. The components of $\mathbf{d}$, i.e., the derivatives of ϕ at node o are obtained by minimizing the Euclidean norm of the error vector $\mathbf{E}$, the ith element of which is defined as

$$E_i = \Delta\phi_i - \sum_{q=1}^{l} \sum_{m=0}^{q} \binom{q}{m} \frac{\Delta x_i^{q-m} \Delta y_i^m}{q!} \frac{\partial^q \phi}{\partial x^{q-m} \partial y^m}. \tag{5}$$

This is equivalent to solving the least-squares problem

$$\mathbf{A}^{\mathrm{T}}\mathbf{Ad} = \mathbf{A}^{\mathrm{T}}\mathbf{b}. \tag{6}$$

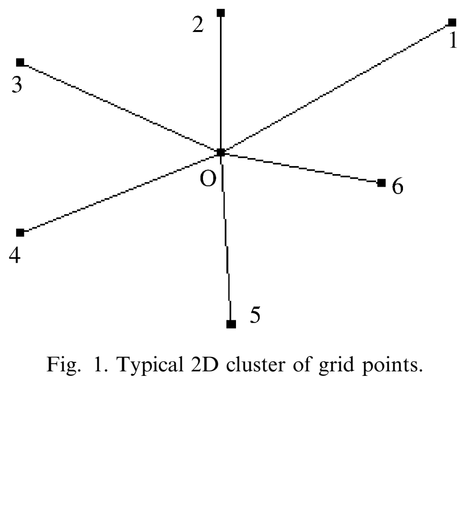


Fig. 1. Typical 2D cluster of grid points.

Because of the fact that the elements of **A** are generated by making use of a polynomial basis, the rank of **A** is $l(l+3)/2$. Therefore, in general, $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ is nonsingular and the solution to the least-squares problem is unique [14]. Though $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ is a nonsingular matrix, a practical implementation of the aforesaid procedure should involve calculation of its condition number. This can be done at a preprocessing stage, because $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ is purely a geometric matrix. A general guiding principle for circumventing the ill-conditioning of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ is to use a stencil of grid points sufficiently larger than the number of unknowns involved in the least-squares procedure. Care should be taken that too large a stencil may smear the local gradients, while a stencil that just satisfies the mathematical requirement may result in the ill-conditioning of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$. More discussions on the preprocessor to be used in the least-squares procedure are presented in Section 6.1.

The accuracy of the least-squares procedure is stated in the following theorem, the proof of which is straight forward and presented in Appendix A.

Theorem 1. *If a given solution ϕ varies smoothly and if discrete values of ϕ are specified at each node, then the least-squares finite difference procedure given by Eq.* (6) *approximates the nth derivative of ϕ, for $n \leqslant l$, to order $h^{l-(n-1)}$, with terms up to lth degree retained in the truncated Taylor series given in Eq.* (2).

It is well known that the Euler equations of gas dynamics are hyperbolic and have distinct directions of information propagation. The least-squares procedure described in this section does not take into account the hyperbolicity of the Euler equations and therefore the numerical scheme based on this procedure can be expected to be unstable. One of the ways to overcome this problem is to enforce upwinding in the numerical procedure. In the subsequent sections we discuss in detail the upwind implementations of the least-squares finite difference procedure.

3. Least-squares finite difference method based on flux vector splitting

Flux vector splitting [29] and flux difference splitting [26] are two well-established strategies for enforcing upwinding while obtaining solutions to Euler equations numerically. In this section we present the use of the least-squares finite difference procedure in conjunction with flux vector splitting strategy. For this purpose we consider 1D Euler equations given by

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} = 0. \tag{7}$$

The 1D grid used for the computation are presented in Fig. 2. Note that the grid points are not uniformly placed. In the flux vector splitting schemes, the flux vector **f** is split into a positive part $\mathbf{f}^{+}$ and a negative part $\mathbf{f}^{-}$. This is given by

$$\mathbf{f} = \mathbf{f}^{+} + \mathbf{f}^{-}. \tag{8}$$

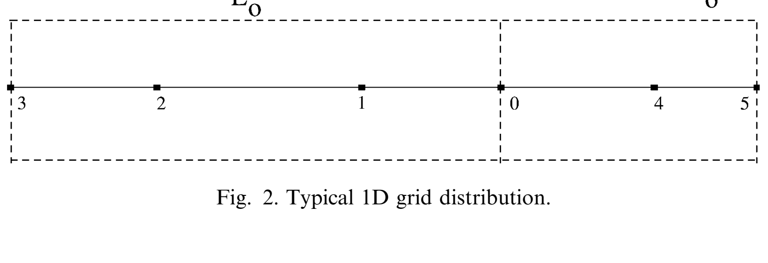


Fig. 2. Typical 1D grid distribution.

The positive flux is said to correspond to the right running waves and the negative flux is said to correspond to the left running waves. Using the split fluxes the 1D Euler equations is recast as

$$\frac{\partial \mathbf{U}}{\partial t}+\frac{\partial \mathbf{f}^{+}}{\partial x}+\frac{\partial \mathbf{f}^{-}}{\partial x}=0. \tag{9}$$

Now, obtaining an upwind least-squares scheme for such an equation is straight forward. The least-squares approximation to the derivatives of positive fluxes is obtained making use of a left stencil of grid points 'L_o' and those of the negative fluxes is obtained from a right stencil of grid points 'R_o.' This is pictorially represented in Fig. 2. Using a linear least-squares fit, the following expressions are obtained for the split flux derivatives:

$$\begin{aligned}\frac{\partial \mathbf{f}^{+}}{\partial x}&=\frac{\sum_{i\in L_0}\Delta \mathbf{f}_i^{+}\Delta x_i}{\sum_{i\in L_0}\Delta x_i^2},\\ \frac{\partial \mathbf{f}^{-}}{\partial x}&=\frac{\sum_{i\in R_0}\Delta \mathbf{f}_i^{-}\Delta x_i}{\sum_{i\in R_0}\Delta x_i^2}.\end{aligned} \tag{10}$$

The least-squares kinetic upwind method (LSKUM) is obtained based on a procedure exactly similar to the one described above, but starting from the Boltzmann equation of the kinetic theory of gases. The readers are referred to [12] for further details.

4. Upwind least-squares finite difference method

In this section we introduce the new methodology as applied to 1D flows. Consider the 1D grid presented in Fig. 3. At the heart of the present methodology is the calculation of an upwind flux at a fictitious interface I associated with the neighboring node i (in line with the cell vertex finite volume schemes) and use this flux in the least-squares formula. Simply stated, for a linear least-squares fit, the expression approximating the flux derivative at o reads

$$f_{x_0}=\frac{\sum_i \Delta F_I \Delta x_I}{\sum_i \Delta x_I^2}, \tag{11}$$

with $\Delta(\cdot)_I=(\cdot)_I-(\cdot)_0$. In the above formula F_I represents the upwind flux calculated at the fictitious interface whose coordinate is given by x_I. The new upwind method differs from the method described in Section 3 in two ways. First, it makes use of the upwind fluxes computed at fictitious interface in the least-squares formula and not the nodal values of the fluxes. The second and more important difference is that it makes use of a global stencil of grid points and not a one sided upwind stencil. This feature on one hand is expected to enhance the accuracy of computation because a smaller stencil of grid points is used compared to the earlier framework, while at the same time considerably reducing the effort required in book keeping. Also, because of the fact that the present strategy does not require a one sided upwind stencil of grid points,

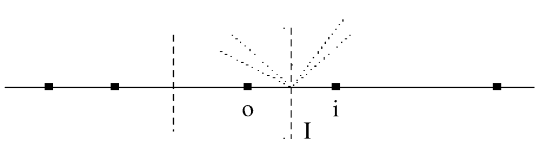


Fig. 3. Use of fictitious interface in the case of 1D problem.

the difficulty in locating the physically relevant neighbors for a given node, particularly near the domain boundaries is completely avoided. In fact this can be a major issue in the flux vector splitting implementation described in the previous section.

Now we demonstrate that the present framework leads to an upwind scheme. This is done based on the theory of Flux difference splitting schemes [26] and later generalized to all upwind schemes. At any given fictitious interface a left state L and a right state R can be defined based on the state variables at corresponding nodes. Using a suitable linearisation procedure the flux difference $\Delta F = F_{\mathrm{R}} - F_{\mathrm{L}}$ can be split into a positive part ΔF^{+} and a negative part ΔF^{-} as follows:

$$\Delta F = \Delta F^{+} + \Delta F^{-}. \tag{12}$$

The flux difference is split in such a way that ΔF^{+} gets its contribution from the right running waves and ΔF^{-} gets its contribution from the left running waves. The interfacial flux in such a case is given by

$$F_I = F_{\mathrm{L}} + \Delta F^{-}, \quad F_I = F_{\mathrm{R}} - \Delta F^{+}. \tag{13}$$

Very often, in finite volume framework, an average of the above two expressions is made use of in the determination of the interfacial flux. This clearly demonstrates that an upwind flux difference between any 'fictitious interface' and node o can be determined using a suitable linearisation procedure. Such a flux difference when applied in the least-squares formula for f_{x_o}, as in Eq. (11), will lead to an upwind least-squares finite difference method (LSFD-U). It is not necessary that such a flux difference should be determined using an explicit use of a linearisation procedure adopted in flux difference splitting schemes. In fact, the flux F_I at the fictitious interface can be determined using any upwind flux formula and the flux difference ΔF_I thus determined would correspond to some hypothetical linearisation step.

5. LSFD-U in 2D

In Section 2 we have already discussed the solution to 2D Euler equations using a least-squares-based update procedure. We had also remarked that such a procedure using a global stencil of grid points, disregarding the hyperbolicity of the Euler equations, can result in an unstable scheme. Here, we extend the idea presented in the previous section to 2D computation.

Consider the 2D Euler equation presented in Eq. (1). A typical 2D cluster of grid points used in LSFD-U is presented in Fig. 4. The state update at o requires discrete approximation to the spatial derivatives $\mathbf{f}_x$ and $\mathbf{g}_y$ at o. In order to determine this using the upwind least-squares finite difference method (LSFD-U), similar to the 1D procedure discussed in Section 4, we introduce a fictitious interface I associated with the neighbor i of the node o. Let $\Delta\vec{r}_I = (\Delta x_I, \Delta y_I)$ represent the vector $\vec{OI}$ and $\widehat{n_I} = (n_{x_I}, n_{y_I})$ represent the unit vector along $\vec{OI}$. The directional flux $F_{\uparrow}$ along $\vec{OI}$ is given by

$$F_{\uparrow} = f n_{x_I} + g n_{y_I}. \tag{14}$$

The interfacial flux $F_{\uparrow I}$ can be determined using any upwind scheme. A k-exact reconstruction procedure (refer Appendix B) is employed to determine the left and right states at the fictitious interface and an upwind flux formula is used to determine the interfacial flux $F_{\uparrow I}$. The flux difference associated with the neighbor i is given by

$$\Delta F_{\uparrow I} = F_{\uparrow I} - F_{\uparrow o}. \tag{15}$$

Now, the task at hand is to recover the gradient of 2D fluxes $\nabla\vec{\mathbf{f}}_o$ and $\nabla\vec{\mathbf{g}}_o$ from the many directional flux difference terms, each of which are essentially uni-dimensional. Expanding the interfacial flux

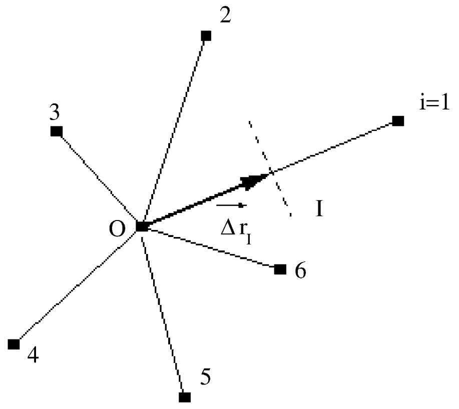


Fig. 4. 2D implementation of LSFD-U.

using a truncated Taylor series, we have the following estimate for the flux difference term given in Eq. (15):

$$\Delta F^e_{\uparrow I} = \frac{1}{|\Delta\vec{r}_I|}\left[\sum_{q=1}^{l}\sum_{m=0}^{q}\binom{q}{m}\frac{\partial^q f}{\partial x^{q-m}\,\partial y^m}\frac{\Delta x_I^{q-m+1}\Delta y_I^m}{q!} + \sum_{q=1}^{l}\sum_{m=0}^{q}\binom{q}{m}\frac{\partial^q g}{\partial x^{q-m}\,\partial y^m}\frac{\Delta x_I^{q-m}\Delta y_I^{m+1}}{q!}\right]. \tag{16}$$

Eq. (16) can be rewritten as

$$\Delta F^e_{\uparrow I} = \frac{1}{|\Delta\vec{r}_I|}\sum_{q=1}^{l}\left[\frac{\partial^q f}{\partial x^q}\frac{\Delta x_I^{q+1}}{q!} + \sum_{m=0}^{q-1}\left\{\binom{q}{m+1}\frac{\partial^q f}{\partial x^{q-(m+1)}\,\partial y^{m+1}} + \binom{q}{m}\frac{\partial^q g}{\partial x^{q-m}\,\partial y^m}\right\}\frac{\Delta x_I^{q-m}\Delta y_I^{m+1}}{q!} + \frac{\partial^q g}{\partial y^q}\frac{\Delta y_I^{q+1}}{q!}\right]. \tag{17}$$

Eq. (17) along with Eq. (15) forms an over determined system of equations in which the unknowns are flux derivatives or their sums. Now we solve for the unknowns using the least-squares procedure. The flux derivatives $\mathbf{f}_x$ and $\mathbf{g}_y$ appearing in the conservation laws form a part of the solution of the least-squares problem. Using those flux derivatives, the state at o can be updated using a suitable time integration procedure.

The least-squares problem is described below. We define vectors $\mathbf{k}_q$ and $\mathbf{d}^e_q$ as follows:

$$\mathbf{k}_q = \frac{1}{q!}\left[\Delta x_I^{q+1} \quad \Delta x_I^q \Delta y_I \quad \ldots \quad \Delta y_I^{q+1}\right], \tag{18}$$

$$\mathbf{d}^e_q = \left[\left(\frac{\partial^q f}{\partial x^q}\right)^e \quad \left(q\frac{\partial^q f}{\partial x^{q-1}\,\partial y} + \frac{\partial^q g}{\partial x^q}\right)^e \quad \ldots \quad \left(\frac{\partial^q g}{\partial y^q}\right)^e\right]^{\mathrm{T}}. \tag{19}$$

In Eq. (19) superscript e over the derivatives are included to indicate that they are numerically estimated values and are not exact. Using Eqs. (18) and (19), Eq. (17) can be recast as

$$\Delta F^e_{\uparrow I} = \frac{1}{|\Delta r_I|}\sum_{q=1}^{l}\mathbf{k}_q \cdot \mathbf{d}^e_q. \tag{20}$$

Now defining an error E_I

$$E_I = \left(\Delta F_{\uparrow I} - \Delta F^e_{\uparrow I}\right)|\Delta\vec{r}_I| \tag{21}$$

and minimizing $\sum_I E_I^2$ with respect to the components of $\mathbf{d}^e_p$, we have the following set of $(p+2)$ equations for any given p:

$$\sum_I E_I \mathbf{k}_p^{\mathrm{T}} = 0 \quad \text{for } 1 \leqslant p \leqslant l. \tag{22}$$

In total for all p we have $l(l+5)/2$ equations and an equal number of unknowns. The system can be cast as

$$\mathbb{A}\mathbf{D}^e_F = \mathbf{B}. \tag{23}$$

The block elements of $\mathbb{A}$ given by A_{pq} is a matrix of dimension $(p+2)\times(q+2)$ and is given by

$$A_{pq} = \frac{1}{p!q!}\begin{bmatrix} \sum \Delta x_I^{p+q+2} & \cdots & \cdots & \cdots & \cdots \\ \sum \Delta x_I^{p+q+1}\Delta y_I & \sum \Delta x_I^{p+q}\Delta y_I^2 & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum \Delta x_I^{q+1}\Delta y_I^{p+1} & \sum \Delta x_I^{q}\Delta y_I^{p+2} & \cdots & \cdots & \sum \Delta y_I^{p+q+2} \end{bmatrix}. \tag{24}$$

The components of $\mathbf{D}^e_F$ and $\mathbf{B}$ represent vectors of dimension $(p+2)$ and are given by

$$\mathbf{D}^e_F = \left[\mathbf{d}_1^{e^{\mathrm{T}}} \quad \mathbf{d}_2^{e^{\mathrm{T}}} \quad \cdots \quad \mathbf{d}_l^{e^{\mathrm{T}}}\right]^{\mathrm{T}}, \tag{25}$$

$$\mathbf{B} = \left[\mathbf{b}_1^{\mathrm{T}} \quad \mathbf{b}_2^{\mathrm{T}} \quad \cdots \quad \mathbf{b}_l^{\mathrm{T}}\right]^{\mathrm{T}}, \tag{26}$$

where $\mathbf{b}_p = \sum_I \Delta F_{\uparrow I}|\Delta\vec{r}_I|\mathbf{k}_p^{\mathrm{T}}$. It is required to solve the system given in Eq. (23) to update the state at any given node. The fact that $\mathbb{A}$ is a symmetric geometric matrix, which can be inverted at a preprocessing stage itself, should be exploited in the practical determination of flux derivatives $\mathbf{f}_x$ and $\mathbf{g}_y$ to required order of accuracy.

5.1. Order of accuracy of LSFD-U

One of the important differences between the present least-squares (LSFD-U) framework and the conventional least-squares framework (presented in Section 2) is that the present framework makes use of the upwind fluxes computed at the fictitious interface I in contrast to the nodal fluxes used in the conventional framework. This is done in order to enforce upwinding. As can be clearly seen, the order of accuracy to which the flux derivatives $\mathbf{f}_{x_o}$ and $\mathbf{g}_{y_o}$ are estimated depends upon, (i) accuracy to which the interfacial flux is estimated (this depends upon the reconstruction procedure described in Appendix B) and (ii) accuracy of the least-squares procedure itself. The following theorem states the order of accuracy of LSFD-U [2,6].

Theorem 2. *If a k-exact reconstruction procedure is employed to determine the directional flux $F_\uparrow$ at the fictitious interface and if a truncated Taylor series of degree l given by Eq.* (17) *is used in the least-squares procedure, then the pth derivatives of the fluxes, for $p \leqslant l$, are determined to* $\mathrm{O}(h^{(l+1)-p})$ *for $k \geqslant l$.*

Proof. In order to prove Theorem 2, to start with, we discuss the order of accuracy least-squares procedure and then discuss its relation to the accuracy associated with the reconstruction procedure.

From Eq. (26) we have

$$\mathbf{b}_p = \sum_I \Delta F_{\uparrow I} |\Delta \vec{r}_I| \mathbf{k}_P^{\mathrm{T}}. \tag{27}$$

Expanding $\Delta F_{\uparrow I}$ appearing in Eq. (27) by Taylor series, we have

$$\mathbf{b}_p = \sum_I \left[\sum_{q=1}^{\infty} \mathbf{k}_q \cdot \mathbf{d}_q \right] \mathbf{k}_p^{\mathrm{T}}, \tag{28}$$

with

$$\mathbf{d}_q = \begin{bmatrix} \frac{\partial^q f}{\partial x^q} & q\frac{\partial^q f}{\partial x^{q-1}\partial y} + \frac{\partial^q g}{\partial x^q} & \dots & \frac{\partial^q g}{\partial y^q} \end{bmatrix}^{\mathrm{T}}. \tag{29}$$

Note that the components of vector $\mathbf{d}_q^e$ represent the numerical estimates to the flux derivatives, the components of vector $\mathbf{d}_q$ represent the exact flux derivatives. The Eq. (28) can be recast as

$$\mathbf{b}_p = \sum_I \left[\sum_{q=1}^{l} \mathbf{k}_q \cdot \mathbf{d}_q \right] \mathbf{k}_p^{\mathrm{T}} + \sum_I \left[\sum_{m=1}^{\infty} \mathbf{k}_{l+m} \cdot \mathbf{d}_{l+m} \right] \mathbf{k}_p^{\mathrm{T}}. \tag{30}$$

The above equation cast in matrix form reads

$$\mathbf{B} = \mathbb{A}\mathbf{D}_{F1} + \mathbb{C}\mathbf{D}_{F2}. \tag{31}$$

The vectors $\mathbf{D}_{F1}$ and $\mathbf{D}_{F2}$ are similar to $\mathbf{D}_F^e$ defined in Eq. (25) and are given by

$$\begin{aligned} \mathbf{D}_{F1} &= \begin{bmatrix} \mathbf{d}_1^{\mathrm{T}} & \mathbf{d}_2^{\mathrm{T}} & \dots & \mathbf{d}_l^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \\ \mathbf{D}_{F2} &= \begin{bmatrix} \mathbf{d}_{l+1}^{\mathrm{T}} & \mathbf{d}_{l+2}^{\mathrm{T}} & \dots & \mathbf{d}_{\infty}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}. \end{aligned} \tag{32}$$

The block elements C_{pm} of matrix $\mathbb{C}$ are defined exactly similar to A_{pq} given in Eq. (24). Introducing Eq. (31) in (23) we have

$$\mathbf{D}_F^e = \mathbf{D}_{F1} + \mathbb{E}\mathbf{D}_{F2}, \tag{33}$$

where $\mathbb{E} = \mathbb{A}^{-1}\mathbb{C}$. With $(A_{pq})^{-1}$ representing the block element constituting $\mathbb{A}^{-1}$ and noting that entries in $A_{pq} \sim \mathrm{O}(h^{p+q+2})$ and $A_{pq} = A_{pq}^{\mathrm{T}}$, we have the entries in $(A_{pq})^{-1} \sim \mathrm{O}(h^{-(p+q+2)})$. Also, the entries in $C_{pm} \sim \mathrm{O}(h^{p+(l+m)+2})$. The leading truncation error corresponds to the case $m = 1$. This gives

$$\mathbf{d}_p^e = \mathbf{d}_p + \mathrm{O}(h^{(l+1)-p}). \tag{34}$$

Thus we see that the least-squares procedure estimates the pth derivatives of the fluxes to $\mathrm{O}(h^{(l+1)-p})$, if exact values of the directional fluxes at the fictitious interface are available. If we represent the directional flux at the fictitious interface obtained using an upwind flux formula by $F_{\uparrow I}^{\mathrm{n}}$ (where superscript n stands for the numerical estimate), we have for a k-exact reconstruction procedure,

$$\Delta F_{\uparrow I}^n = \Delta F_{\uparrow I} + \mathrm{O}(h^{k+1}). \tag{35}$$

Because of the approximation associated with the determination of the interfacial flux $F_{\uparrow I}$, Eq. (34) gets modified as

$$\mathbf{d}_p^e = \mathbf{d}_p + T_1 + T_2, \tag{36}$$

where $T_1 \sim \mathrm{O}(h^{(l+1)-p})$ and $T_2 \sim \mathrm{O}(h^{(k+1)-p})$. □

Remark 1. The case $p = 1$, corresponding to the first derivatives, is of interest to us and we have for $k \geqslant l$

$$\mathbf{d}_1^e = \mathbf{d}_1 + O(h^l).$$

Therefore we clearly see that the LSFD-U procedure demands that $k \geqslant l$, in order to retain the accuracy of the least-squares formula and we have to at least choose 1-exact (linear) reconstruction procedure for the LSFD-U scheme to be consistent.

Remark 2. The use of weights $w_i \sim 1/d_{oi}^n$, where $d_{oi} = [(x_o - x_i)^2 + (y_o - y_i)^2]^{1/2}$ in the least-squares procedure, does not alter any of the arguments pertaining to the accuracy of the LSFD-U procedure.

With respect to the first remark made above, it should be mentioned that it is indeed possible to construct a LSFD-U procedure corrected for consistency employing the nodal values of the fluxes themselves. This leads to many interesting possibilities, including its utility in implicit LSFD-U and limiting, the details of which are not of direct consequence to the subject matter presented in this paper and therefore would be presented elsewhere.

5.2. Variants of LSFD-U

In this section we consider two useful variants of LSFD-U based on a linear least-squares fit. For a linear fit, Eq. (17) becomes

$$\Delta F_{\uparrow I}^e = \frac{1}{\Delta \vec{r}_I}\left[\frac{\partial f}{\partial x}\Delta x_I^2 + \left(\frac{\partial f}{\partial y} + \frac{\partial g}{\partial x}\right)\Delta x_I \Delta y_I + \frac{\partial g}{\partial y}\Delta y_I^2\right]. \tag{37}$$

The least-squares minimisation procedure results in Eq. (23) which in the present case is given by

$$\begin{bmatrix} \sum \Delta x_I^4 & \sum \Delta x_I^3 \Delta y_I & \sum \Delta x_I^2 \Delta y_I^2 \\ \sum \Delta x_I^3 \Delta y_I & \sum \Delta x_I^2 \Delta y_I^2 & \sum \Delta x_I \Delta y_I^3 \\ \sum \Delta x_I^2 \Delta y_I^2 & \sum \Delta x_I \Delta y_I^3 & \sum \Delta y_I^4 \end{bmatrix} \begin{bmatrix} f_{xo} \\ f_{yo} + g_{xo} \\ g_{yo} \end{bmatrix} = \begin{bmatrix} \sum \Delta F_I |\Delta \vec{r}_I| \Delta x_I^2 \\ \sum \Delta F_I |\Delta \vec{r}_I| \Delta x_I \Delta y_I \\ \sum \Delta F_I |\Delta \vec{r}_I| \Delta y_I^2 \end{bmatrix}. \tag{38}$$

This LSFD-U procedure can be suitably modified based on physical and mathematical arguments and this results in the $\vec{q}$-variant and the θ-variant of the procedure. These are described in the subsequent section.

5.2.1. $\vec{q}$-variant: LSFD-U($\vec{q}$)

In this method we locally rotate the co-ordinate in such a way that one of the axes coincides with the streamline direction (refer to Fig. 5). Now we consider the conservation laws on the new coordinate system. At any given point o, with respect to the new co-ordinate, the fluxes normal to the stream-wise direction involve only pressure terms. We expect that the flow in the close vicinity of o would also be dominated by this feature. It is well known in CFD that the pressure terms are elliptic in nature. In fact, such a coordinate rotation can be considered as an effective means to separate the convective part of the flux which is hyperbolic in nature from the pressure part which is essentially elliptic. Calculations on such rotated coordinates are not uncommon. As examples, we can cite the work of Ghosh and co-workers [11], where the coordinate rotation is used as a means for reducing dissipation in the scheme, the work of Jameson [13] on "rotated difference scheme" for introducing upwinding in the potential flow calculation and the work of Levy et al. [15] for obtaining grid independent solution. Such coordinate system permits the use of a global

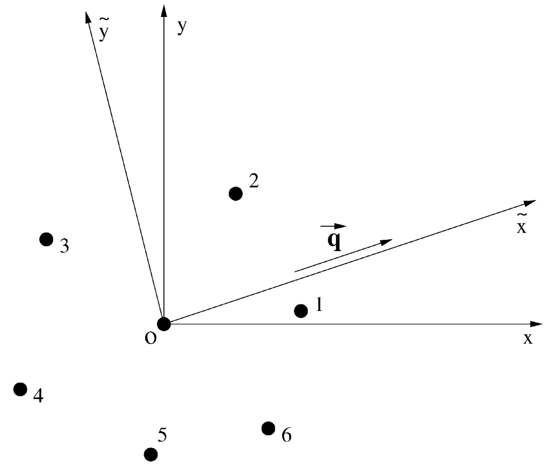


Fig. 5. Rotated co-ordinate for LSFD-U($\vec{q}$).

stencil of grid points in conjunction with the conventional least-squares procedure, described in Section 2, in the computation of the derivatives of the flux component normal to the stream-wise direction. That is if $\tilde{\mathbf{f}}$ and $\tilde{\mathbf{g}}$ represent the fluxes along new coordinate directions $\tilde{x}$ (the stream-wise direction) and $\tilde{y}$, respectively, as a consequence of the ellipticity of pressure terms, $\nabla\vec{\tilde{\mathbf{g}}}_o$ can be computed by a conventional least-squares procedure using a global stencil of grid points. Treating the $\nabla\vec{\tilde{\mathbf{g}}}_o$ as known, we define a new flux difference term in place of the one defined in Eq. (37),

$$\Delta\tilde{F}_{\uparrow I} = \Delta F_{\uparrow I} - \frac{1}{|\Delta\vec{r}_I|}\left(\tilde{g}_{\tilde{x}_o}\Delta\tilde{x}_I\Delta\tilde{y}_I + \tilde{g}_{\tilde{y}_o}\Delta\tilde{y}_I^2\right). \tag{39}$$

In the new framework an expression for the estimate of the flux difference $\Delta\tilde{F}^e_{\uparrow I}$ would read

$$\Delta\tilde{F}^e_{\uparrow I} = \frac{1}{|\Delta\vec{r}_I|}\left(\tilde{f}_{\tilde{x}_o}\Delta\tilde{x}_I^2 + \tilde{f}_{\tilde{y}_o}\Delta\tilde{x}_I\Delta\tilde{y}_I\right). \tag{40}$$

Eq. (40) along with (39) defines an overdetermined system of equations with unknowns $\tilde{\mathbf{f}}_{\tilde{x}_o}$ and $\tilde{\mathbf{f}}_{\tilde{y}_o}$, which can be solved using the least-squares procedure.

5.2.2. θ-variant: LSFD-U(θ)

This method also involves a local rotation of the coordinate. In this case we transform the conservation equation to the new co-ordinate system. It should be remarked that in the present method we continue to work with x and y momentum fluxes, but with respect to a local rotated coordinate $(\tilde{x},\tilde{y})$. On the new co-ordinate Eq. (37) can be recast as

$$\Delta F^e_{\uparrow I} = \frac{1}{|\Delta\vec{r}_I|}\left[\tilde{f}_{\tilde{x}_o}\Delta\tilde{x}_I^2 + \left(\tilde{f}_{\tilde{y}_o} + \tilde{g}_{\tilde{x}_o}\right)\Delta\tilde{x}_I\Delta\tilde{y}_I + \tilde{g}_{\tilde{y}_o}\Delta\tilde{y}_I^2\right]. \tag{41}$$

Now we look for a co-ordinate system where the flux derivative sum $\tilde{\mathbf{f}}_{\tilde{y}_o} + \tilde{\mathbf{g}}_{\tilde{x}_o}$ vanishes. It can be easily shown that a coordinate system rotated at an angel θ given by

$$\theta = \frac{1}{2}\tan^{-1}\left[\frac{f_y + g_x}{f_x - g_y}\right] \tag{42}$$

would satisfy the aforesaid constraint. The derivatives of the fluxes used in the estimation of θ are obtained using the conventional least-squares procedure described in Section 2 using a global stencil of grid points. On the new rotated local co-ordinate it is possible to define modified flux difference, similar to Eq. (39),

$$\Delta F^{e}_{\uparrow I} = \Delta F_{\uparrow I} - \frac{1}{|\Delta \vec{r}_I|}\left(\tilde{f}_{\tilde{y}_o} + \tilde{g}_{\tilde{x}_o}\right)\Delta\tilde{x}_I\Delta\tilde{y}_I. \tag{43}$$

In this case the estimate of the modified flux difference $\Delta\tilde{F}^{e}_{\uparrow I}$ would read

$$\Delta F^{e}_{\uparrow I} = \frac{1}{|\Delta \vec{r}_I|}\left(\tilde{f}_{\tilde{x}_o}\Delta\tilde{x}_I + \tilde{g}_{\tilde{y}_o}\Delta\tilde{y}_I\right). \tag{44}$$

Eq. (44) along with (43) defines an over determined system of equations with unknowns $\tilde{\mathbf{f}}_{\tilde{x}_o}$ and $\tilde{\mathbf{g}}_{\tilde{y}_o}$, which can be solved using the least-squares procedure. A practical implementation of the θ-variant of LSFD-U procedure, would involve using a numerical estimate for the flux derivative sum $(\tilde{\mathbf{f}}_{\tilde{y}_o} + \tilde{\mathbf{g}}_{\tilde{x}_o})$ appearing in Eq. (43), though this is expected to be very small, and can be obtained using a conventional least-squares procedure described in Section 2.

It is worthwhile to make the following comments, particularly, in the context of extending the LSFD-U(θ) to 3D computations [2]. The coordinate transformation involved in the present method is essentially a principal axis transformation well known in the field of continuum mechanics. To demonstrate this we introduce a second-order tensor of flux derivatives,

$$\prod_{ij} = \frac{1}{2}\left[\frac{\partial f_i}{\partial x_j} + \frac{\partial f_j}{\partial x_i}\right].$$

In the above equation f_1, f_2 and f_3 represent the flux components along the coordinate directions x_1, x_2 and x_3. It can be easily seen that $\prod$ represent a symmetric tensor and our interest lies in the computation of $\partial f_i/\partial x_i$, the trace of $\prod$, which is invariant with respect to rotation of coordinates. We know from the theory of matrices, for any symmetric tensor there exists an orthogonal set of axes (called principal axes) with respect to which the tensor may be cast in diagonal form. The orthogonal set of axes are obtained by solving the eigenvalue problem

$$\left(\prod_{ij} - \lambda\delta_{ij}\right)n_j = 0 \quad (i = 1, 2, 3),$$

where λ represent the principal value and the associated eigenvector represents the principal axis. Therefore in LSFD-U(θ) the job is to locally determine the principal axes, transform the conservation laws to the principal axis coordinate system and solve the equations on the transformed plane.

5.2.3. Accuracy of the variants of LSFD-U

In Section 5.1 we have already discussed order of accuracy of a general LSFD-U procedure. Based on the discussions presented in Section 5.1, it is clear that, in the $\vec{q}$ and θ variants of the LSFD-U, second-order estimates of the modified flux differences (given by Eqs. (39) and (43)) should be used for the scheme to be consistent. This requires the use of a linear reconstruction procedure in conjunction with a first order estimate of the flux derivatives appearing in the expression for the modified flux difference.

6. Results and discussion

This section presents the results of various computations done using LSFD-U and its variants. These cases were selected accordingly to test the ability of LSFD-U over wide range of Mach number, ranging from Low Subsonic ($M = 0.1$) to High Supersonic ($M = 3.0$) Mach numbers. Also, various interfacial flux formulas based on Flux Vector Splitting and Flux Difference Splitting were made use of to show the flexibility, the LSFD-U procedure offers. The ability of different interfacial flux formulas in handling different flow situations is effectively exploited in the present least-squares computations. In the present computations, the points required for LSFD-U state update are obtained from unstructured triangulated grid. This facilitates comparison of LSFD-U results with those of cell-vertex finite volume. It is important to note that the connectivity used for the LSFD-U procedure is obtained independent of the connectivity given by the grid generator. This is done at a preprocessor level and is explained below.

6.1. Preprocessor

One of the important components of LSFD-U computation is the use of a preprocessor to generate the grid data. The least-squares procedure involves inversion of geometric matrices, which have been shown to be nonsingular in Section 2. In addition, it is also required that these matrices are not ill-conditioned. This constraint is extremely crucial because the flux derivatives obtained using the least-squares procedure are directly used in the state update formula. Therefore, the choice of right connectivity for the grid points is the key to success of any least-squares procedure. In the present case the domain around any given point is divided into a specified number of sectors (user defined) and the points are included into the connectivity from each of these sectors based on a proximity criterion. This defines a primary set of neighbours. If the condition number requirement is not satisfied, a point from a secondary set of neighbours is included into the connectivity, if it results in a improvement in the condition number of the resulting geometric matrix. A general guiding principle for the selection of neighbours is to uniformly distribute them around a given point. This can be understood from the modified equation analysis, the details of which would be presented elsewhere. Also, it should be remembered that the problems pertaining to the conditioning of the geometric matrix is also encountered in the case of least-squares-based reconstruction procedure as applied to finite volume computations [3]. Given the fact that the geometric matrix is positive definite, a simple strategy like diagonal preconditioning greatly alleviates this problem (S. Nikhil, N. Balakrishnan, A new migratory memory algorithm (MMA) for implicit finite volume solvers, AIAA J., submitted). Also, it should be mentioned that an effective search algorithm can become a prerequisite while handling large grid data.

6.2. Boundary condition

The following boundary conditions are used in the present computations:

(1) Strong boundary condition developed by Balakrishnan and Fernandez [4] is used on the solid wall. The use of such boundary condition developed for cell vertex finite volume schemes is very relevant for least-squares-based update procedure.
(2) At the far field, free-stream conditions are enforced for supersonic flow and flow induced by a point vortex of a equivalent strength placed at mid-chord of an airfoil is specified for subsonic and transonic flows [28].
(3) Supersonic inflow and outflow boundary conditions are used for the case of internal flows.

6.3. Validative computations

The point distribution for LSFD-U state update, in the present computations, are obtained from unstructured triangulated grid [22]. The results from LSFD-U and its variants are compared with that of cell vertex finite volume on the same grid. A linear reconstruction of the primitive variable [3,7] in conjunction with Venkatakrishnan limiter [30] is used throughout. Four-stage Runge–Kutta time stepping is used for time integration. The convergence is assumed after 6 decades of fall in relative residue. The flow variables in present computations are nondimensionalized with the free-stream values. The density and velocity are nondimensionalized with their respective free-stream values. The pressure is nondimensionalized as $p/\rho_\infty U_\infty^2$. We also make use of an entropy like variable $s = p/\rho^\gamma$ in the contour plots, in order to dem-

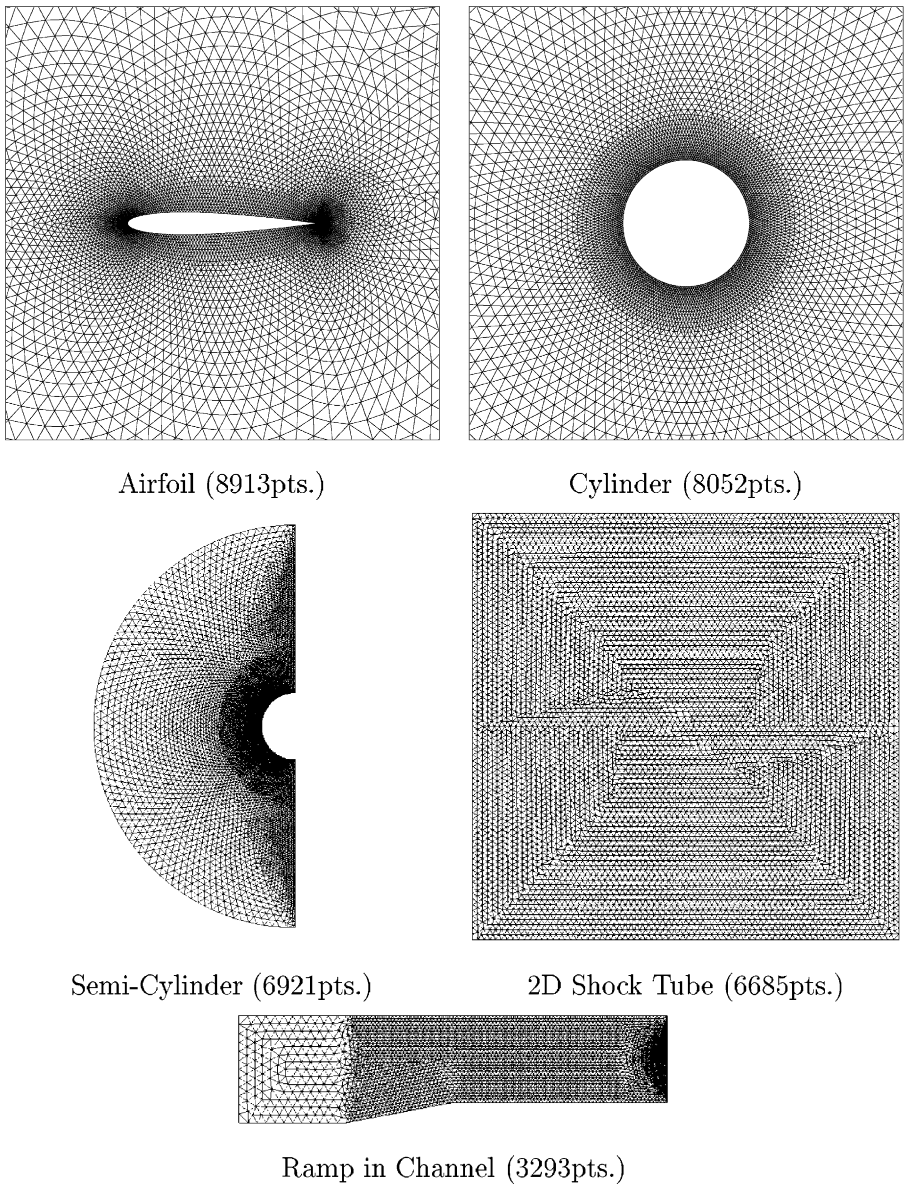


Fig. 6. Various grids used in present computations.

Table 1

Test case	Grid		Numerical flux formula
	No. of nodes	No. of nodes on the wall	
1. Potential flow past a cylinder ($M_\infty = 0.1$)	8052[a]	200	Roe [26]
2. Transonic flow past NACA 0012 ($M_\infty = 0.85$, $\alpha = 1.0°$)	8913[a]	198	AUSM [19]
3. Supersonic flow past semi-cylinder ($M_\infty = 3.0$)	6921[b]	120	KFVS [18]

[a] The far-field = 10 chords.
[b] The far-field = 3 radii.

onstrate the spurious entropy generation by the schemes. The test cases considered, grid details (see Fig. 6) and the numerical flux formula used are presented in Table 1.

6.3.1. Low subsonic flow over cylinder

In this case flow past a cylinder for a free-stream Mach number of 0.1 is simulated. This is a potential flow case for which exact solution is available. This case tests the ability of the new algorithm in capturing flow features pertaining to low speed flows. The C_p plot and the pressure contours obtained using various algorithms are presented in Figs. 7 and 8, respectively. It should be mentioned that the top–bottom asymmetry (though very minimal) as observed in the C_p plot is a consequence of the asymmetry of the unstructured mesh employed for the computation. Also, the C_p plot and the pressure contours reveal that the left–right symmetry of the pressure field is reproduced remarkably well. In fact, there is a marginal loss of left–right symmetry in the Mach contour (not shown) due to entropy layer generated at wall.

6.3.2. Transonic flow over NACA 0012

In this case transonic flow ($M_\infty = 0.85$) past NACA 0012 airfoil at an angle of attack of 1° is simulated. This is a standard AGARD [1] test case in the transonic regime. The C_L and C_D obtained using LSFD-U

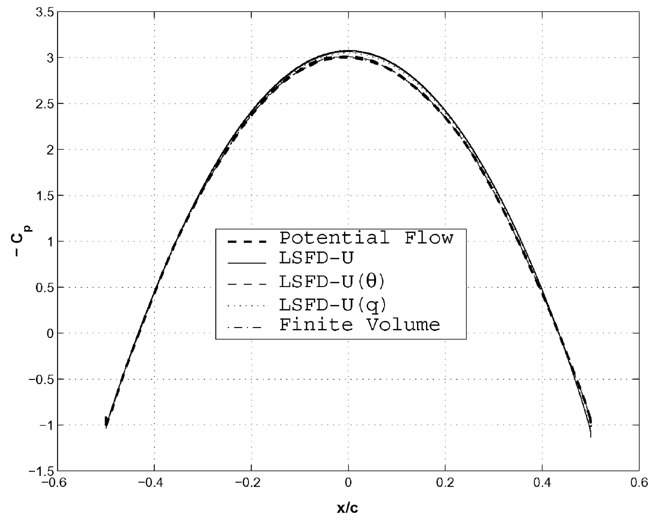
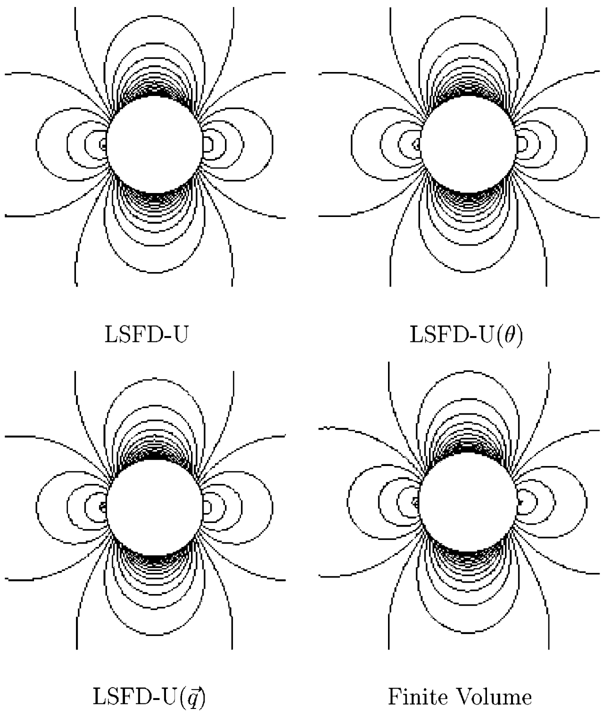


Fig. 7. C_p plot for flow past cylinder at $M = 0.1$ and $\alpha = 0°$.



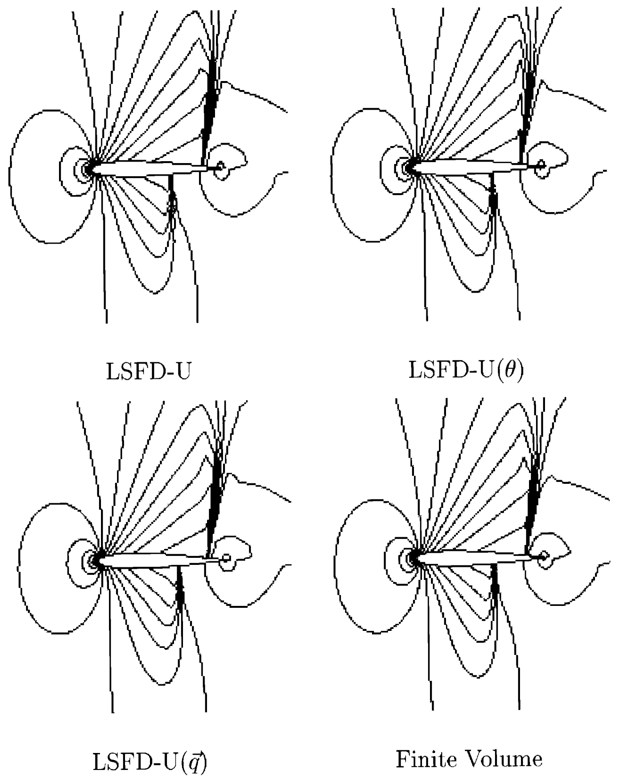
[min = 69.8676, Δ = 0.1076, max = 72.0193]

Fig. 8. Pressure contours. Subsonic flow past cylinder $M_\infty = 0.1$.

and its variants compared with those obtained using cell vertex finite volume scheme and the AGARD values in Table 2. The Mach contours, entropy contours and C_p plots are presented in Figs. 9–11, respectively. The results clearly bring out the fact that the jumps across the top and bottom shocks, their locations and wall pressure distributions are all captured accurately by the present framework. In Table 3 we present the theoretical and computational entropy jump across shock on the top wall. The theoretical jump is calculated using normal shock relation for the upstream Mach number before the shock on the top wall. The jumps computed are within and around 5% of theoretical value. Accurate computation of exact jump across the shock is related to the conservation property of the numerical scheme. The percentage errors reported in Table 3 is a quantitative evidence in this regard. It should be emphasized that the results presented clearly reveal the ability of LSFD-U in retaining the nicer features of the AUSM flux formula as applied to finite volume computations.

Table 2
C_L and C_D values for various method using AUSM flux splitting

	LSFD-U	LSFD-U(θ)	LSFD-U($\vec{q}$)	Finite volume	AGARD
C_L	0.3881	0.3611	0.3746	0.3912	0.330–0.389
C_D	0.0556	0.0523	0.0566	0.0555	0.0464–0.0590



[min = 0.0, Δ = 0.0738, max = 1.4758]

Fig. 9. Mach contours. Transonic flow past NACA 0012 $M_\infty = 0.85$, $\alpha = 1°$.

6.3.3. High supersonic flow over semi-cylinder

Results pertaining to supersonic flow ($M_\infty = 3.0$) past a semi-cylinder are presented here. This case tests the robustness of the algorithm. The shock standoff distance in this case can be calculated using Ambrosio and Wortman [9] correlation,

$$\frac{\Delta}{R} = 0.386\exp(4.67/M^2).$$

The total pressure variation along the symmetry line as obtained from the computation is compared with the exact values in Fig. 13. In plotting the exact total pressure variation the shock standoff distance, as obtained from the above formula, is made use of. The pressure contours obtained using LSFD-U, LSFD-U(θ) and cell vertex finite volume procedures are presented in Fig. 12. The results obtained demonstrate the robustness of the present framework and are very encouraging. It should be stated that it was not possible to obtain results using the $\vec{q}$-variant. As we had indicated earlier, in the case of LSFD-U($\vec{q}$) numerical dissipation to ensure stability of the time integration procedure is brought in through the upwinding of the stream-wise fluxes. In flows, such as the one considered in this test case, where the fluid is brought rapidly to rest close to the stagnation point, the numerical dissipation resulting from the upwinding of the stream-wise derivatives is not adequate to stabilize the scheme and this results in lack of robustness.

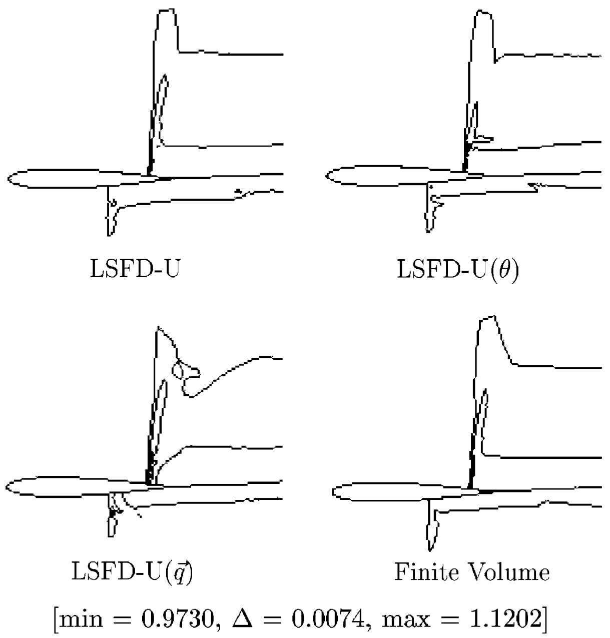


Fig. 10. Entropy contour. Transonic flow past NACA 0012 $M_\infty = 0.85$, $\alpha = 1°$.

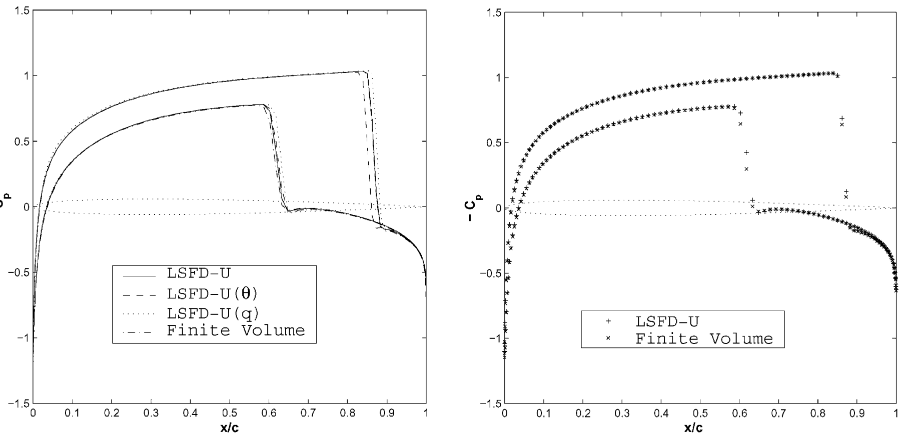


Fig. 11. C_p plot. Transonic flow past NACA 0012 $M_\infty = 0.85$, $\alpha = 1°$.

Table 3
Wall entropy jump across the top normal shock

	M_1	ΔS_{comp}	ΔS_{theo}	% relative error
LSFD-U	1.4392	0.0220	0.0214	2.80
LSFD-U(θ)	1.4351	0.0198	0.0209	5.26
LSFD-U($\vec{q}$)	1.4623	0.0246	0.0242	1.65
Finite volume	1.4373	0.0220	0.0212	3.58

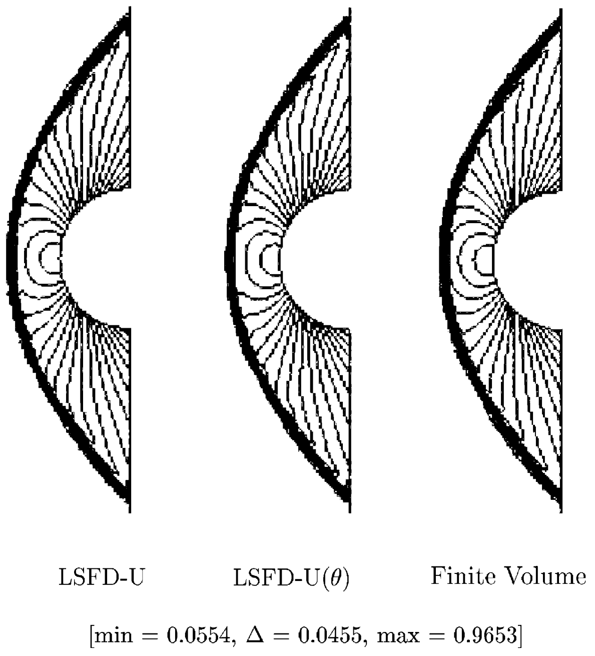


Fig. 12. Pressure contours. High supersonic flow past semi-cylinder $M_\infty = 3.0$.

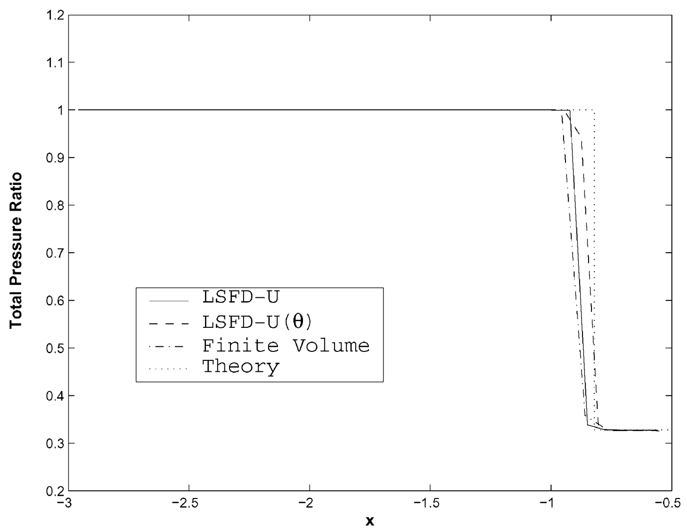


Fig. 13. Symmetry line total pressure ratio variation. $M_\infty = 3.0$ flow past semi-cylinder.

6.4. Conservation study

One of the very important aspect of numerical schemes based on a generalized finite difference procedure like LSFD-U is that it is difficult to prove the conservation of the scheme as in the case of finite volume

procedure. In order to empirically demonstrate the conservation of the LSFD-U framework, two test problems were chosen. They are:
(1) Supersonic flow past a 10.75° ramp in a channel.
(2) Unsteady shock tube problem cast in two dimensions.

The conservation is demonstrated by systematically employing a series of numerical experiments. Starting from a coarse triangular grid, finer grids are obtained by successive refinement wherein each triangle is divided into four triangles. The LSFD-U state update is operated on the resulting arbitrary point distribution.

Case 1. Supersonic flow past ramp in a channel. This test case, in contrast to the external flow cases seen in the previous sections, is an ideal candidate for studying the conservation property of a numerical procedure. In this case, conservation demands that the mass inflow into the channel should be same as the outflow at steady state. Any difference in the mass flow can be indicative of the numerical source/sink

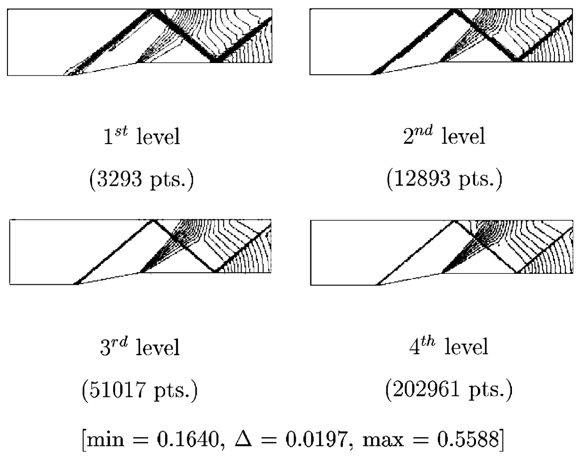


Fig. 14. Pressure contours. Supersonic ($M_{\text{inlet}} = 2.0$) flow past 10.75° ramp in a channel for 4 levels of grids.

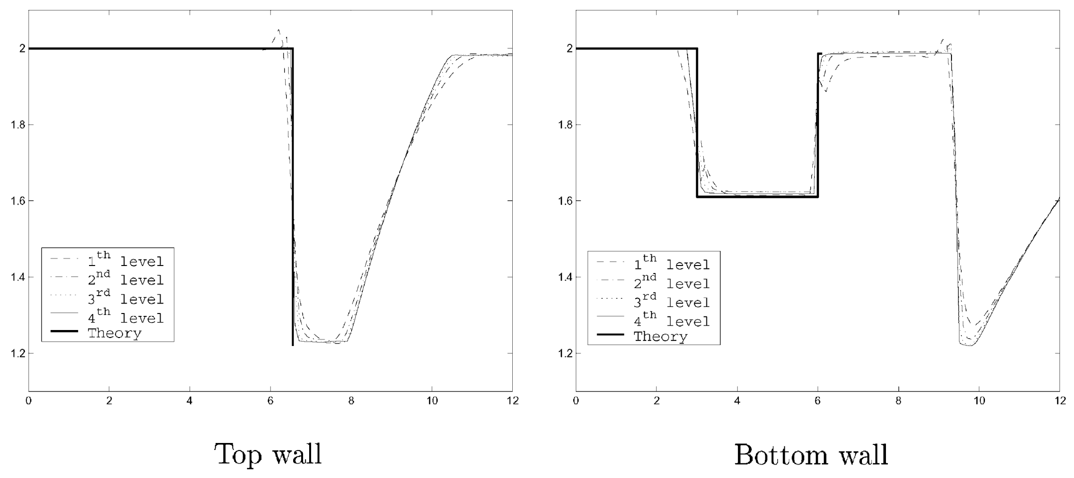


Fig. 15. Wall Mach number distribution obtained on various levels of grid refinement.

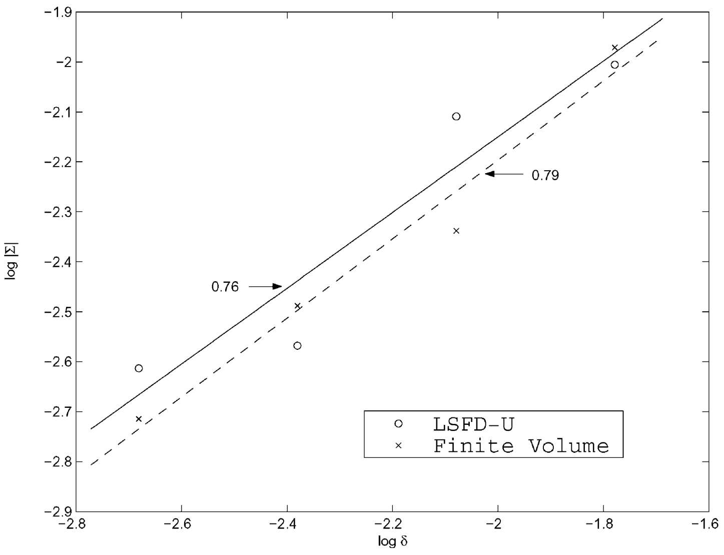


Fig. 16. Estimate of conservation for various levels of refinements.

associated with the algorithm. We employ this difference $\sum$ as an indicator of loss of conservation of the numerical procedure. It should be remarked that in the case of finite volume schemes, $\sum$ would be comparable to the residue defined on mass (at the time of declaring convergence), because of the telescopic collapse of the flux integral on the volume interface. The logarithmic value of $\sum$ is plotted against $\log\delta$, where δ is mean point spacing, in Fig. 16. The slope of the best line fit (solid line) for LSFD-U reveals

$$\sum \simeq \mathrm{O}(\delta) \quad \text{as } \delta \to 0.$$

The above result amply demonstrates the conservation property of the LSFD-U method, as the value of $\sum$ (indicative of loss in conservation) decays to zero at least as fast as the δ does. An interesting comment with regard to the use of cell vertex finite volume procedure in conjunction with a nonconservative strong wall boundary treatment [4] can be made here. In such a case the mass balance $\sum$ obtained using the finite volume computation matches with that obtained using the LSFD-U procedure, as depicted in Fig. 16. At the same time, the use of finite volume scheme with conservative mirror boundary condition on the wall yields $\sum$ comparable to the residue at the time of declaring convergence. In Fig. 14, the pressure contours obtained using four different grids, resulting from a successive refinement procedure described above, are presented. It can be clearly seen that the flow features captured on the coarser grid become sharper on the finer ones. In Fig. 15 the computed wall Mach numbers are compared with the theoretical values, in which the theoretical values are plotted only up to the reflection of oblique shock on the top and expansion on the bottom walls, respectively. These results clearly demonstrate, both qualitatively and quantitatively, the accuracy of LSFD-U.

In addition to conservation, we demonstrate the robustness of the present methodology in handling grid distortions by the use of random grid. The grid shown in Fig. 17 is generated by placing points randomly in both the coordinate directions in such a way that it is possible to recover regular triangles from the points for cell vertex finite volume computations. The Mach contours for both LSFD-U and cell vertex finite volume procedure, presented in Fig. 17, show good matching. The computed wall Mach numbers are compared with the exact values in Fig. 18. The figure amply demonstrates the ability of LSFD-U in capturing shocks of right strength at right locations apart from other continuous flow features.

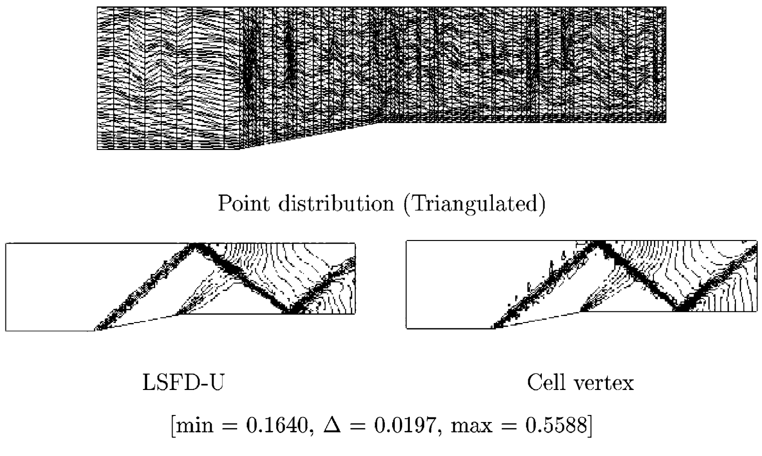

Point distribution (Triangulated)
LSFD-U
Cell vertex
[min = 0.1640, Δ = 0.0197, max = 0.5588]

Fig. 17. Grid and Mach contours obtained on a random point distribution (2844 pts.)

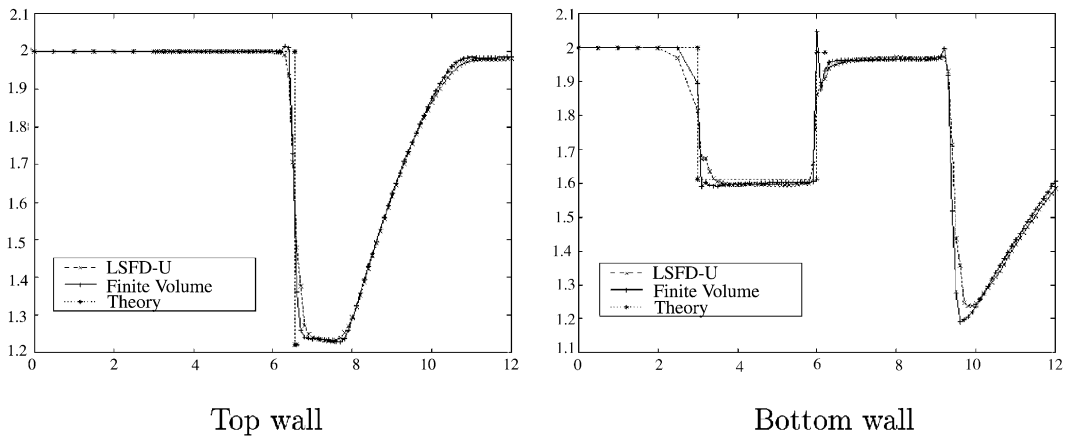


Fig. 18. Wall Mach number distribution obtained on the random grid.

Case 2. Sod's shock tube problem [27]. For an unsteady flow case, accurate computation of shock speed requires satisfaction of conservation property by the numerical scheme. Having demonstrated the conservation property of LSFD-U for a steady problem in previous section we take up the unsteady case in present. For this purpose the standard Sod's shock tube problem was cast in two dimensions and solved on an arbitrary distribution of points obtained using an unstructured mesh generator. Like in previous case, here too the finer grids were obtained from coarser grids by successive refinement. The density contours at time $t = 7.5 \times 10^{-4}$, obtained using LSFD-U on different grids, are compared with the cell vertex finite volume (shown only for first grid, i.e., 6685 pts) solution in Fig. 19. From the figure one can clearly see the three features namely, shock, contact discontinuity and expansion are captured accurately. The mid-section density variation from present computations are plotted with analytical solution in Fig. 20, to reinforce the above observation. The features, as seen from Figs. 19 and 20, approach the exact solution with successive grid refinements. The logarithm of L_2 norm of error, as compared to the theoretical solution, of mid-section density variation is plotted against the logarithm of mean point spacing in Fig. 21. Again, from the slope of the best line fit, it is observed

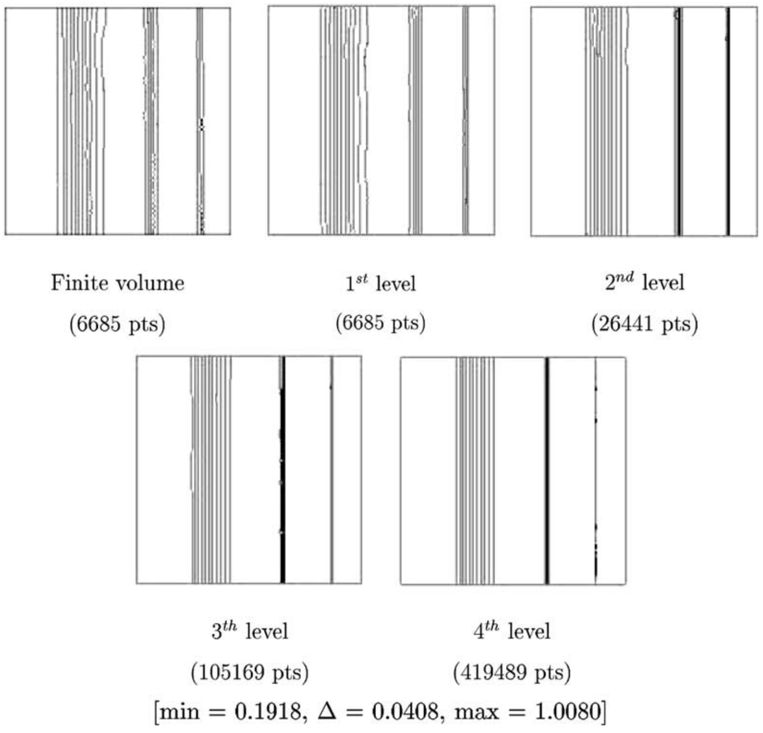


Fig. 19. Density contours. Sod's shock tube problem in 2D.

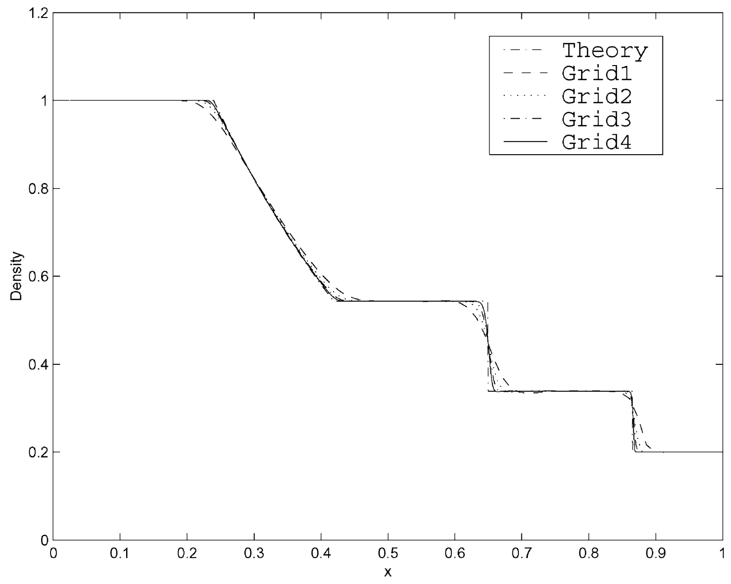


Fig. 20. Mid-section density plot. Sod's shock tube problem in 2D.

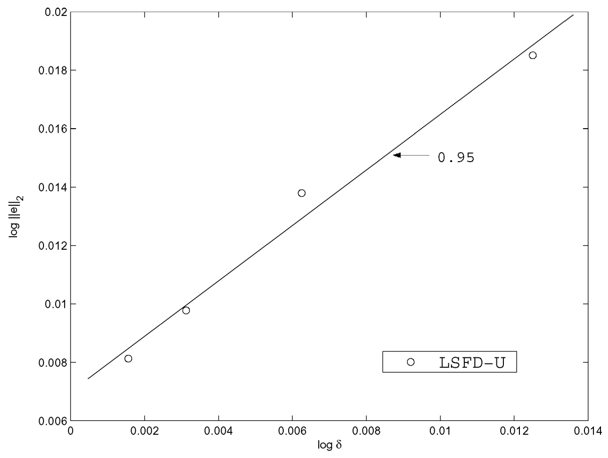


Fig. 21. Mid-section L_2 norm of error for various levels of grid refinement.

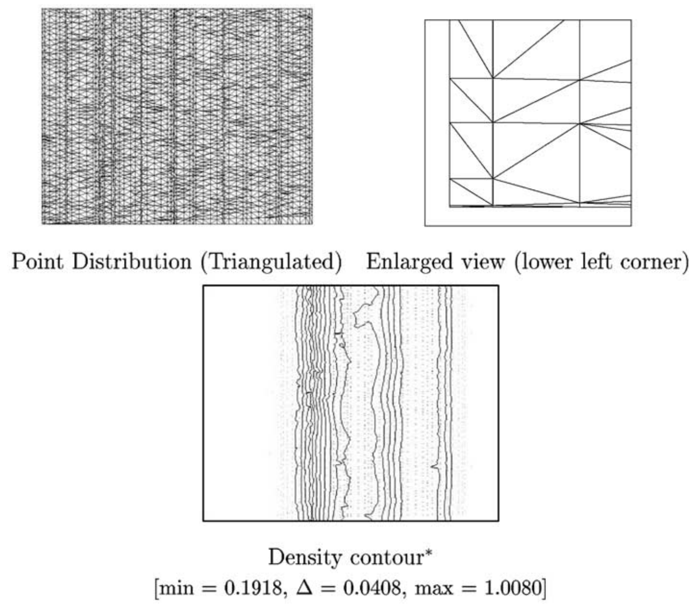

Point Distribution (Triangulated)
Enlarged view (lower left corner)
Density contour*
[min = 0.1918, Δ = 0.0408, max = 1.0080]

Fig. 22. Sod's shock tube problem in 2D on random points distribution, 3599 pts. * Shown with points distribution in background.

$$||e||_2 \simeq \mathrm{O}(\delta) \quad \text{as } \delta \to 0.$$

It is similar to the observation made for steady case in previous section. The L_2 norm of error decreasing as δ in the limit $\delta \to 0$ indicates the computed solution approaches the exact in the limit.

Similar to the previous case, we again choose to demonstrate the robustness of LSFD-U framework for an unsteady flow problem, using a randomly generated grid shown in Fig. 22. The density contours, in Fig. 22, clearly demonstrates the ability of present framework to capture various flow features. The midsection

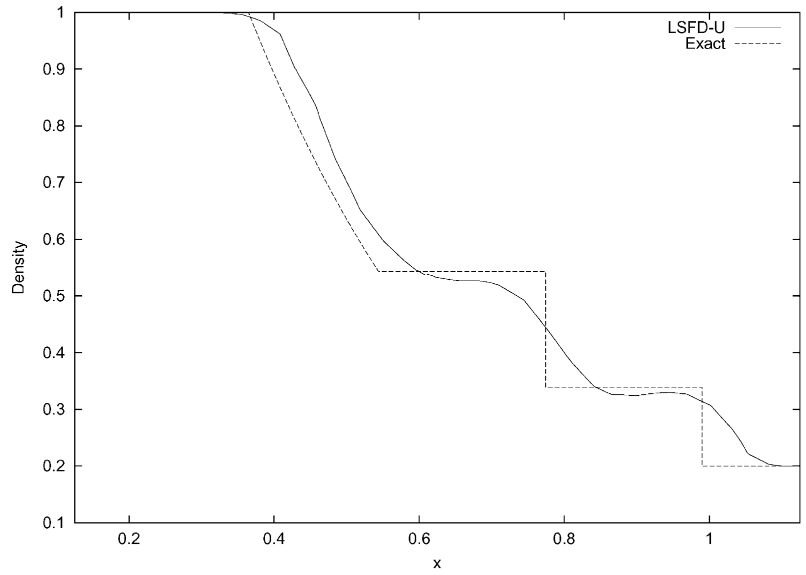


Fig. 23. Mid-section density plot. Sod's shock tube problem in 2D on random points distribution.

density distribution compared with exact solution, in Fig. 23, amply demonstrates the strength of the present framework to operate on highly distorted grids. The degree of grid distortion involved in this computation is brought out in Fig. 22, where an enlarged view of the left bottom corner of the computational domain is presented. It should be remarked here that the authors were unable to obtain a finite volume solution on this grid.

7. Conclusions

A new upwind generalized finite difference scheme, LSFD-U has been developed. This scheme has the flexibility of being used on any type of grids or even a random distribution of points. It falls under the class of Meshless Schemes. This new framework makes use of global stencil of points compared to other methods, like LSKUM, which uses one sided upwind stencil of grid points for state update. This is one of the important advantages of LSFD-U. Such a stencil, being more compact, is expected to enhance the accuracy. A means to construct a LSFD-U procedure of specified order of accuracy has been presented. It is demonstrated that for a first-order estimation of flux derivatives $\mathbf{f}_x$ and $\mathbf{g}_y$ for state update at any node, we need to have at least 1-exact reconstruction for LSFD-U to be consistent. The LSFD-U and its variants have been tested over a wide range of Mach numbers and their solutions compare well with those obtained using cell vertex finite volume employing a linear reconstruction procedure. Also, in these computations, the flexibility of LSFD-U in using various upwind flux formula (both Flux Vector Splitting and Flux Difference Splitting) has been demonstrated. Though it is theoretically difficult to prove the conservation of schemes based on generalized finite difference procedure, in present case we choose an empirical approach to demonstrate the conservation of LSFD-U. Numerical experiments performed for the steady internal flow case and the unsteady flow problem are clearly indicative of the conservation of the present least-squares procedure. It should also be remarked that the computational effort involved in LSFD-U state update is comparable with that of cell vertex finite volume update, in all the cases considered in this paper.

Appendix A

Proof of Theorem 1. Here we prove the order of accuracy of a general least-squares procedure described in Section 2, in particular for a 1D case. The 2D generalisation of this proof follows exactly the same lines as the procedure described for LSFD-U in Section 5.1.

Suppose that discrete values of function $\phi(x)$ are available at nodes of the computational domain. Consider a node i in the neighbourhood of o. The truncated Taylor series estimate of ϕ_i is given by

$$\phi_i^e = \phi_o + \sum_{q=1}^{l} \frac{\Delta x_i^q}{q!} \frac{\mathrm{d}^q \phi}{\mathrm{d}x^q}. \tag{A.1}$$

Defining an associated error $E_i = \phi_i - \phi_i^e$ and minimizing the Euclidean norm of the error results in

$$\sum_i \sum_{q=1}^{l} \frac{\Delta x_i^p}{p!} \frac{\Delta x_i^q}{q!} \left(\frac{\mathrm{d}^q \phi}{\mathrm{d}x^q}\right)^e = \sum_i \Delta\phi_i \frac{\Delta x_i^p}{p!} \quad \text{for } p = 1 \text{ to } l. \tag{A.2}$$

The superscript e for the derivatives indicate that they are approximate numerical estimates. Eq. (A.2) represents a system of l equations with l unknowns and can be recast as

$$\mathbb{A}\mathbf{D}_\phi^e = \mathbf{B}, \tag{A.3}$$

with

$$A_{pq} = \sum_i \frac{\Delta x_i^p}{p!} \frac{\Delta x_i^q}{q!},$$

$$\mathbf{D}_\phi^e = \left[\frac{\mathrm{d}\phi}{\mathrm{d}x}\bigg|^e \frac{\mathrm{d}^2\phi}{\mathrm{d}x^2}\bigg|^e \cdots \frac{\mathrm{d}^l\phi}{\mathrm{d}x^l}\bigg|^e \right]^{\mathrm{T}},$$

$$\mathbf{b}_p = \sum_i \Delta\phi_i \frac{\Delta x_i^p}{p!}.$$

In the above equation, A_{pq} and $\mathbf{b}_p$ represent the elements of matrix $\mathbb{A}$ and vector $\mathbf{B}$, respectively, and $\mathbf{D}_\phi^e$ is the vector of estimated derivatives of ϕ. Order of accuracy to which the elements of $\mathbf{D}_\phi^e$ are determined is obtained by expanding $\Delta\phi_i$ appearing in $\mathbf{b}_p$ in Taylor series. We have

$$\mathbf{b}_p = \sum_i \left\{ \sum_{q=1}^{l} \frac{\Delta x_i^q}{q!} \frac{\mathrm{d}^q\phi}{\mathrm{d}x^q} + \sum_{m=1}^{\infty} \frac{\Delta x_i^{l+m}}{(l+m)!} \frac{\mathrm{d}^{l+m}\phi}{\mathrm{d}x^{l+m}} \right\} \frac{\Delta x_i^p}{p!}. \tag{A.4}$$

Eq. (A.4) can be recast as

$$\mathbf{B} = \mathbb{A}\mathbf{D}_{\phi,1} + \mathbb{C}\mathbf{D}_{\phi,2}, \tag{A.5}$$

with

$$C_{pm} = \sum_i \frac{\Delta x_i^p}{p!} \frac{\Delta x^{(l+m)}}{(l+m)!},$$

$$\mathbf{D}_{\phi,1} = \left[\frac{\mathrm{d}\phi}{\mathrm{d}x} \quad \frac{\mathrm{d}^2\phi}{\mathrm{d}x^2} \quad \cdots \quad \frac{\mathrm{d}^l\phi}{\mathrm{d}x^l} \right]^{\mathrm{T}}$$

and

$$\mathbf{D}_{\phi,2} = \left[\frac{\mathrm{d}^{l+1}\phi}{\mathrm{d}x^{l+1}} \quad \frac{\mathrm{d}^{l+2}\phi}{\mathrm{d}x^{l+2}} \quad \cdots \quad \frac{\mathrm{d}^{\infty}\phi}{\mathrm{d}x^{\infty}} \right]^{\mathrm{T}}.$$

Introducing (A.5) in (A.3), we have

$$\mathbb{A}\mathbf{D}^e_\phi = \mathbb{A}\mathbf{D}_{\phi,1} + \mathbb{C}\mathbf{D}_{\phi,2}.$$

Premultiplying by $\mathbb{A}^{-1}$ we have,

$$\mathbf{D}^e_\phi = \mathbf{D}_{\phi,1} + \mathbb{E}\mathbf{D}_{\phi,2},$$

with $\mathbb{E} = \mathbb{A}^{-1}\mathbb{C}$ and E_{pm} an element of matrix $\mathbb{E}$ is given by, $E_{pm} = \sum_{q=1}^{l} (A_{pq})^{-1} C_{qm}$ where $(A_{pq})^{-1}$ represents the elements of $\mathbb{A}^{-1}$. The order of accuracy to which $\mathrm{d}^p\phi/\mathrm{d}x^p|^e$ is determined depends upon the order of leading truncation error term, which corresponds to the case $m = 1$. Noting that $\mathbb{A}$ is a symmetric matrix, with $A_{pq} \sim \mathrm{O}(h^{p+q})$, $(A_{pq})^{-1} \sim \mathrm{O}(h^{-(p+q)})$ and $C_{pm} \sim \mathrm{O}(h^{p+l+m})$, we have

$$\left.\frac{\mathrm{d}^p\phi}{\mathrm{d}x^p}\right|^e = \frac{\mathrm{d}^p\phi}{\mathrm{d}x^p} + \mathrm{O}(h^{l+1-p}). \qquad \square$$

Appendix B

The reconstruction algorithm employed in this method is similar to the procedure employed in the case of cell vertex finite volume computations [3]. The procedure is briefly presented here for the sake of completeness.

Assume that discrete solution value $\phi(x, y)$ is available at any node o. We have

$$\phi_o = \phi(\vec{r}_o).$$

The objective of the reconstruction procedure is to represent the solution variation in the neighbourhood of o (i.e., is to reconstruct the information lost during a discrete state update) by a polynomial function $\phi^k_o(\vec{r})$of degree k, satisfying the following constraints:

(1) The polynomial should take the discrete solution value ϕ_o at node, i.e.,

$$\phi^k_o(\vec{r}_o) = \phi_o.$$

(2) The polynomial should represent polynomial functions of degree $l \leqslant k$ exactly.

In the present work a least-squares-based linear reconstruction procedure has been employed. As per this procedure, the solution variation in the neighbourhood of o is given by

$$\phi^1_o = \phi_o + \nabla\phi_o \cdot \Delta\vec{r},$$

where $\Delta\vec{r} = \vec{r} - \vec{r}_o$ and the least-squares formula for the gradients are given by

$$\phi_{xo} = \frac{\|\Delta y\|^2(\Delta x, \Delta\phi) - (\Delta x, \Delta y)(\Delta y, \Delta\phi)}{\|\Delta x\|^2\|\Delta y\|^2 - (\Delta x, \Delta y)^2},$$

$$\phi_{yo} = \frac{\|\Delta x\|^2(\Delta y, \Delta\phi) - (\Delta x, \Delta y)(\Delta x, \Delta\phi)}{\|\Delta x\|^2\|\Delta y\|^2 - (\Delta x, \Delta y)^2},$$

with $\Delta(\cdot) = (\cdot)_i - (\cdot)_o$, $\|\cdot\|^2$ representing the Euclidean norm and (a, b) representing the inner product of two vector quantities. It can be easily seen that such a polynomial exactly recovers both the constant state and a linear function.

References

[1] AGARD, Test cases for inviscid flow field methods, AR 211, Report of Fluid Dynamics Panel Working Group 07, 1986.
[2] N. Balakrishnan, New least squares based finite difference methods, 99 FM 9, Department of Aerospace Engineering, IISc, Bangalore, 1999.
[3] N. Balakrishnan, S.M. Deshpande, Reconstruction on Unstructured Meshes with Upwind Solvers, Proceedings of the First Asian CFD Conference, Hong Kong, 1995.
[4] N. Balakrishnan, G. Fernandez, Wall boundary conditions for inviscid compressible flows on unstructured meshes, Int. J. Numer. Methods Fluids 28 (1998) 1481–1501.
[5] N. Balakrishnan, C. Praveen, A new upwind least squares finite difference scheme (LSFD-U) for Euler equations of gas dynamics, in: Finite Volume for Complex Applications – II, Hermes Science, Paris, 1999.
[6] N. Balakrishnan, D. Sridar, On a generalised finite difference framework and its order of accuracy, in: N. Satofuka (Ed.), Computational Fluid Dynamics 2000, Springer, Berlin, 2000.
[7] T.J. Barth, Higher order solution of the Euler equations on unstructured grids using quadratic reconstruction, AIAA Paper No. 90-0013, 1990.
[8] T.J. Batina, A gridless Euler/Navier–Stokes solution algorithm for complex aircraft applications, AIAA Paper No. 93-0333, 1993.
[9] F.S. Billig, Shock-wave shape around spherical and cylindrical-nosed bodies, J. Spacecraft 4 (6) (1967) 822–823.
[10] K.C. Chung, A generalised finite difference method for heat transfer problems of irregular geometries, Numer. Heat Transfer 4 (1981) 345–357.
[11] S.M. Deshpande, A.K. Ghosh, J.C. Mandal, Least squares weak upwind methods for Euler equations, 89 FM 4, Department of Aerospace Engineering, IISc, Bangalore, 1989.
[12] A.K. Ghosh, Robust least squares kinetic upwind method for inviscid compressible flows, Ph.D. Thesis, Department of Aerospace Engineering, IISc, Bangalore, 1996.
[13] A. Jameson, Iterative solutions of transonic flows over airfoils and wings including flows at Mach 1, Comm. Pure Appl. Math. 27 (1974) 282–309.
[14] L.W. Johnson, R.D. Riess, J.T. Arnold, Introduction to Linear Algebra, Addison-Wesley, Reading, MA, 1989.
[15] D.W. Levy, K.G. Powell, B. van Leer, An implementation of grid independent upwind scheme for Euler equations, AIAA Paper No. 89-1931-CP, 1989.
[16] H. Lin, S.N. Atluri, The meshless local Petrov–Galerkin (MLPG) method for solving incompressible Navier–Stokes equations, CMES 2 (2) (2001) 117–142.
[17] R.W. MacCormack, A perspective on a quarter century of CFD research, AIAA Paper No. 93-3291-CP, 1993.
[18] J.C. Mandal, S.M. Deshpande, Kinetic flux vector splitting for Euler equations, Comp. Fluids 23 (2) (1994) 447–478.
[19] Meng-Sing Liou, C.J. Steffen Jr., A new flux splitting scheme, J. Comput. Phys. 107 (1993) 23–39.
[20] K. Morinishi, An implicit gridless type solver for the Navier–Stokes equations, CFD J. 9 (1) (2000).
[21] K. Morinishi, Effective accuracy and conservation consistency of gridless type solver, in: N. Satofuka (Ed.), Computational Fluid Dynamics 2000, Springer, Berlin, 2000.
[22] J.D. Müller, On triangles and flow, Ph.D. Thesis, The University of Michigan, 1996.
[23] Nobuatsu Tanaka, Development of a highly accurate interpolation method for mesh-free flow simulation I. Integration of gridless, particle and CIP methods, Int. J. Numer. Methods Fluids 30 (1999) 957–976.
[24] C. Praveen, S.M. Deshpande, Rotationally invariant gridless upwind method for Euler equations, 2001 FM 08, Department of Aerospace Engineering, IISc, Bangalore, 2001.
[25] R. Löhner, C. Sacco, E. Oñate, S. Idelsohn, A finite point method for compressible flow, Int. J. Numer. Methods Engrg. 53 (2002) 1765–1779.
[26] P.L. Roe, Approximate Riemann solver, parameter vectors and difference schemes, J. Comput. Phys. 43 (1981) 357–372.
[27] G.A. Sod, A survey of several finite difference methods for system of nonlinear hyperbolic conservation laws, J. Comput. Phys. 27 (1978) 1–31.
[28] J.L. Thomas, M.D. Salas, Far-field boundary condition for transonic lifting solutions to the Euler equations, AIAA Paper No. 85-0020, 1985.

[29] B. van Leer, in: Flux Vector Splitting for Euler Equations, Lecture Notes in Physics, vol. 170, Springer, New York, Berlin, 1982.

[30] V. Venkatakrishnan, Convergence to steady state solutions of the Euler equations on unstructured grids with limiters, J. Comput. Phys. 118 (1995) 120–130.

[31] Wing Kam Liu, Sukky Jun, Yi Fei Zhang, Reproducing Kernel particle methods, Intr. J. Numer. Methods Fluids 20 (1995) 1081–1106.