Computers & Fluids 40 (2011) 118–138

Contents lists available at ScienceDirect

Computers & Fluids

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

Turbulent flow computations on a hybrid cartesian point distribution using meshless solver LSFD-U

N. Munikrishna, N. Balakrishnan *

Computational Aerodynamics Laboratory, Department of Aerospace Engineering, Indian Institute of Science, Bangalore 560 012, India

ARTICLE INFO

Article history:
Received 23 November 2009
Received in revised form 16 August 2010
Accepted 19 August 2010
Available online 6 September 2010

Keywords:
Meshless solver
Viscous discretization
Positivity
Cartesian grid
Turbulent flows

ABSTRACT

This paper may be considered as a sequel to one of our earlier works pertaining to the development of an upwind algorithm for meshless solvers. While the earlier work dealt with the development of an inviscid solution procedure, the present work focuses on its extension to viscous flows. A robust viscous discretization strategy is chosen based on positivity of a discrete Laplacian. This work projects meshless solver as a viable cartesian grid methodology. The point distribution required for the meshless solver is obtained from a hybrid cartesian gridding strategy. Particularly considering the importance of an hybrid cartesian mesh for RANS computations, the difficulties encountered in a conventional least squares based discretization strategy are highlighted. In this context, importance of discretization strategies which exploit the local structure in the grid is presented, along with a suitable point sorting strategy. Of particular interest is the proposed discretization strategies (both inviscid and viscous) within the structured grid block; a rotated update for the inviscid part and a Green-Gauss procedure based positive update for the viscous part. Both these procedures conveniently avoid the ill-conditioning associated with a conventional least squares procedure in the critical region of structured grid block. The robustness and accuracy of such a strategy is demonstrated on a number of standard test cases including a case of a multi-element airfoil. The computational efficiency of the proposed meshless solver is also demonstrated.

1. Introduction

Generating good quality grids, particularly for computing viscous flow past complex configurations still remains a great challenge. Though the recent advancements made in the cartesian mesh based calculations [1–5] basically address this issue, the cartesian mesh algorithms are yet to mature to a level where accurate resolution of turbulent boundary layers using RANS methodologies is possible [5] for complex configurations. It is in this context the developments in the meshless solvers [6–15] become important. The meshless solvers require only a distribution of points for the solution algorithms to operate and not a discretization of the computational domain in the classical sense of finite volume or finite element methodologies. Generating a distribution of points is expected to be a lot easier compared to generating grids, say for a finite volume computation [12]. In spite of this supposed advantage of the meshless solvers, the progress in this area has been rather slow, particularly considering the fact that the earliest work on the generalized finite differences appeared in the year 1981 [16]. Subsequent attempts in the development of the meshless solvers were mostly towards inviscid flow computations [6,7,9]. In our earlier works we have systematically developed the Upwind Least Squares based Finite Difference method (LSFD-U) as applied to inviscid flows [8,9]. The present work involving the extension of these ideas to viscous flows can be considered as a sequel to our earlier work [9]. Interestingly, recent application of meshless solvers for solving realistic industrial problems involving store separation [17,18], clearly brings out the higher levels of acceptability of such procedures amongst the industrial community. In spite of several of these developments in the use of meshless solvers for inviscid flow analysis, there have been very few attempts in solving RANS based turbulent flow equations [10,11,19,20]. Even in such attempts [10,11], the absence of skin friction profile (an important parameter in evaluating viscous flow algorithms) in the presented results, reveals the necessity for further developments in this area. In fact, Refs. [19,20] highlight the difficulties associated with obtaining an accurate skin friction profile using meshless RANS computations. Therefore, this paper focuses on computing turbulent flows using meshless solvers.

Two pertinent questions, necessarily to be answered in the development of meshless solvers are: (a) how do we generate the point distribution? (b) is the algorithm robust enough to work on a given distribution of points?

To answer the first question, it is important to realize that all the meshless computations reported so far have employed conventional mesh generators to provide the required point distribution.

* Corresponding author. Tel.: +91 8022933029; fax: +91 8023600134.
E-mail address: nbalak@aero.iisc.ernet.in (N. Balakrishnan).

0045-7930/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compfluid.2010.08.017

This approach may have advantages for certain class of problems [21] and can be used for demonstrating the efficacy of a newly developed meshless algorithm [9]. But, in general, this approach is not attractive, because it compromises the most fundamental advantage in the use of meshless solvers, that generating a point cloud in the computational domain is easier as compared to generating a grid. Unless a generic point generation strategy is evolved, it is not possible to completely harness the advantages of using a meshless solver. In our view, cartesian meshes provide the most natural way to obtain the point distribution for meshless solvers. It is well known that the use of hybrid cartesian mesh provides most reasonable means to perform RANS based turbulent flow computations [2,3]. In the context of finite volume solvers, such an approach suffers from two major drawbacks, one of small cut cells and the other of handling the non-conformal interface between the cartesian and structured grid blocks [5]. These issues are irrelevant for the meshless solvers requiring only a distribution of points. The earliest work recognizing this aspect of meshless solvers is due to Morinishi [11], which uses a point distribution from hybrid cartesian mesh and employs a meshless update for all these points. In this work, attempts have been made to compute turbulent flow past multi-element airfoils; the results presented pertain to only wall pressure distribution and not skin friction. In the present work, we evolve an objective strategy for utilizing the point distribution obtained employing a cartesian grid framework for RANS based turbulent flow computations. Akin to the work of Morinishi [11], all the points in the computational domain are updated using the meshless solver.

By the way of answering the second of the questions listed above, it should be remarked that the generic least squares based differencing procedures commonly used in meshless solvers run into difficulties when used for point distribution required for resolving turbulent boundary layers in a RANS computation. In this connection, we have also evolved simple and robust discretization procedures exploiting the local structure in the point distribution. In doing so, we have ensured that the generality of the meshless solvers is not compromised. To the best of our knowledge, this work represents the first of its kind in demonstrating the capability of the meshless solvers in solving turbulent flow past complex aerodynamic configurations, wherein the skin friction data are also presented.

The paper is organized as follows. The LSFD-U procedure is presented in Section 2. Point generation strategy is explained in Section 3. In Section 4, the discretization procedures evolved for LSFD-U RANS solver are discussed. In Section 5, numerical results are presented and the concluding remarks are made in Section 6.

2. LSFD-U procedure

The primary aim of this section is to introduce, to an uninitiated reader, the LSFD-U based inviscid discretization procedure we had evolved in our earlier works [8,9,22] and a generalized finite difference procedure for viscous discretization, proposed as part of the present work, based on positivity analysis. The specific details of the discretization procedure, particularly in the context of the point generation strategy proposed in Section 3, will be taken up in Section 4.

2.1. Governing equations

The non-dimensional form of the compressible Reynolds-Averaged Navier–Stokes equation can be expressed in the conservative form as follows:

$$\frac{\partial W}{\partial t}+\frac{\partial (f+F)}{\partial x}+\frac{\partial (g+G)}{\partial y}=0 \tag{1}$$

where

$$W=[\rho \quad \rho u \quad \rho v \quad e]^T,$$
$$f=[\rho u \quad \rho uu+p \quad \rho vu \quad (e+p)u]^T,$$
$$g=[\rho v \quad \rho uv \quad \rho vv+p \quad (e+p)v]^T,$$
$$F=\left[0-\tau_{xx}-\tau_{xy}-\left(u\tau_{xx}+v\tau_{xy}-q_x\right)\right]^T,$$
$$G=\left[0-\tau_{xy}-\tau_{yy}-\left(u\tau_{xy}+v\tau_{yy}-q_y\right)\right]^T.$$

Here W represent the vector of conserved variables and f, g are inviscid fluxes. The terms F and G represent viscous fluxes where, the shear stress and heat conduction terms are given by,

$$\tau_{xx}=\frac{(\mu+\mu_t)}{Re_\infty}\left(2\frac{\partial u}{\partial x}-\frac{2}{3}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right),\quad \tau_{xy}=\frac{(\mu+\mu_t)}{Re_\infty}\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right),$$
$$\tau_{yy}=\frac{(\mu+\mu_t)}{Re_\infty}\left(2\frac{\partial v}{\partial y}-\frac{2}{3}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right),$$
$$q_x=-\frac{(k+k_t)}{(\gamma-1)M_\infty^2 Pr_\infty Re_\infty}\left(\frac{\partial T}{\partial x}\right),\quad q_y=-\frac{(k+k_t)}{(\gamma-1)M_\infty^2 Pr_\infty Re_\infty}\left(\frac{\partial T}{\partial y}\right).$$

The equation of state is given as,

$$p=\frac{\rho T}{\gamma M_\infty^2}. \tag{2}$$

The flow variables are non-dimensionalized by the free stream density ρ_∞, free stream velocity U_∞, free stream temperature T_∞, viscosity μ_∞, thermal conductivity k_∞, and a reference length L. In the above non-dimensional form of the equations, M_∞ denotes freestream Mach number; Pr_∞ represents freestream Prandtl number. Freestream Reynolds number is defined as $Re_\infty=\frac{\rho_\infty U_\infty L}{\mu_\infty}$. The laminar viscosity and thermal conductivity are determined using Sutherland's law [23] with a free stream temperature 273°k. The terms μ_t and k_t represent turbulent viscosity and turbulent thermal conductivity respectively.

2.2. Least squares procedure

Here for the sake of completion we present the conventional least squares procedure [8,9] for approximating the spatial derivatives. A typical 2-D cluster of grid points is shown in Fig. 1. We introduce an arbitrary function ϕ, the discrete values of which are known at grid points. The function value in the neighbourhood of node i (say at node j) can be approximated using a truncated Taylor series

$$\phi_j=\phi_i+\sum_{q=1}^{l}\sum_{m=0}^{q}\binom{q}{m}\frac{\Delta x_j^{q-m}\Delta y_j^m}{q!}\frac{\partial^q\phi}{\partial x^{q-m}\partial y^m}, \tag{3}$$

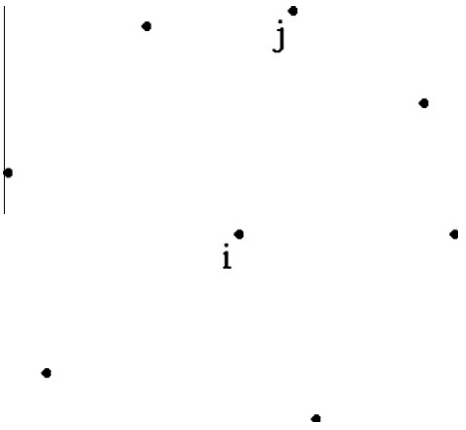


Fig. 1. Typical 2-D cluster.

where $\Delta(\cdot)_j = (\cdot)_j - (\cdot)_i$. Let n represent the number of neighbours of node i. Eq. (3) represents an over determined system of equations for $n > l(l+3)/2$ and can be solved using method of least squares. The error associated with neighbour j is given by,

$$E_j = \Delta\phi_j - \sum_{q=1}^{l}\sum_{m=0}^{q}\binom{q}{m}\frac{\Delta x_j^{q-m}\Delta y_j^m}{q!}\frac{\partial^q\phi}{\partial x^{q-m}\partial y^m}. \tag{4}$$

The derivatives of ϕ at node i can be determined by minimising $\sum_j E_j^2$. The resulting system of equations can be expressed in matrix-form as,

$$\mathbf{Ax} = \mathbf{b}, \tag{5}$$

where $\mathbf{A}$ is a geometric matrix, $\mathbf{x}$ is derivative vector and $\mathbf{b}$ is RHS vector. The accuracy of the derivatives obtained using least squares procedure is presented in the following remark [9].

Remark 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. (5) approximates the nth derivative of ϕ, for $n \leqslant l$, to order $h^{l-(n-1)}$, with terms upto lth degree retained in the truncated Taylor series given in Eq. (3).

It is evident from the above remark that a linear least squares procedure results in first order accurate gradients and a quadratic least squares procedure results in second order accurate gradients. In Sections 2.3 and 2.4, we introduce least squares based discretization procedures for a general point distribution, for both the inviscid and viscous flux derivatives. After introducing the point generation strategy in Section 3, the specifics of the discretization procedures employing the local structure in the point distribution would be introduced.

2.3. Inviscid flux discretization

Here we present details of discretization of the inviscid flux derivatives. A typical 2-D cluster of grid points, for any update at node i, along with its neighbours j, is shown in Fig. 2a. The task at hand is to approximate the flux derivatives at node i using the generalized finite difference procedure based on the method of least squares [7,8]. We have demonstrated in our earlier work [8,9], the importance of introducing a fictitious interface J at the mid-point of the ray ij, for the purpose of enforcing upwinding. In the present work, upwinding is done along the coordinate directions [22], unlike our earlier implementation, where upwinding is effected along the ray ij. The procedure simply involves, determination of the split fluxes along the coordinate directions at the fictitious interface, employing a suitable reconstruction procedure. This is pictorially depicted in Fig. 2b. The preference of this variant of LSFD-U over the other three variants discussed in Ref. [9] is because of the additional robustness the present implementation offers [24]. In Ref. [9], we have shown that the order of accuracy of the LSFD-U procedure depends on the accuracy to which the interfacial fluxes are determined and the accuracy of the least squares procedure itself. In the present work, we have employed a linear reconstruction of the primitive variables in the determination of the fluxes at the fictitious interface in conjunction with a weighted linear least squares procedure for the determination of the flux derivatives, resulting in a consistent discretization procedure (refer to theorem 2, in Ref. [9]). The discrete approximation for the flux derivatives read,

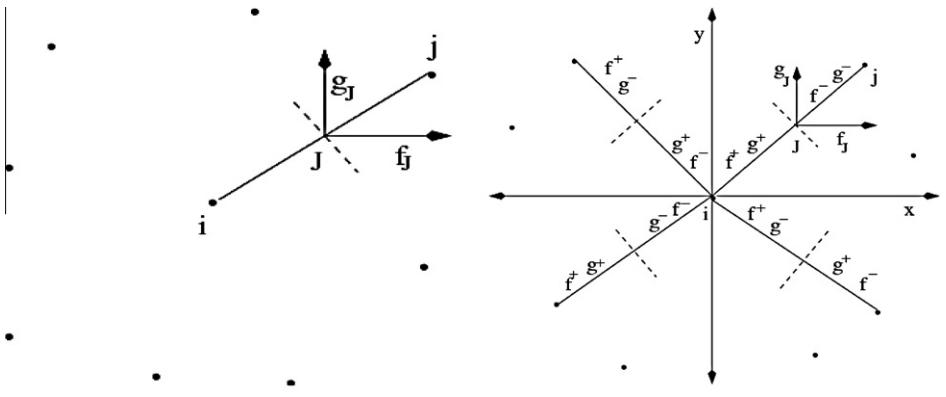
$$f_{x_i} = \frac{\sum_J w_J\Delta y_J^2\sum_J w_J\Delta f_J\Delta x_J - \sum_J w_J\Delta x_J\Delta y_J\sum_J w_J\Delta f_J\Delta y_J}{\sum_J w_J\Delta x_J^2\sum_J w_J\Delta y_J^2 - \left(\sum_J w_J\Delta x_J\Delta y_J\right)^2}, \tag{6}$$

$$g_{y_i} = \frac{-\sum_J w_J\Delta x_J\Delta y_J\sum_J w_J\Delta g_J\Delta x_J + \sum_J w_J\Delta x_J^2\sum_J w_J\Delta g_J\Delta y_J}{\sum_J w_J\Delta x_J^2\sum_J w_J\Delta y_J^2 - \left(\sum_J w_J\Delta x_J\Delta y_J\right)^2}, \tag{7}$$

where f and g are the inviscid flux vectors. In Eqs. (6) and (7), it is physically meaningful to use weights $w_J \sim |\cos\theta_J|$ or $w_J \sim \frac{\cos^2\theta_J}{l_J^2}$, where θ_J is the angle made by the ray ij, with the coordinate direction along which the flux derivatives are being determined [25]. In spite of this observation, in our experience, use of unit weights is found to be adequate for establishing a robust solution procedure and therefore, unit weights have been employed for generating the solutions presented in this work.

2.4. Viscous flux discretization

As indicated before, in spite of the success of the meshless solvers in computing inviscid flows [9,17], their extension to RANS computations has been non-trivial [19,20]. Two major hurdles in the RANS computations as applied to meshless solvers for a generalized point distribution are the non-positivity of the viscous discretization procedure [19] and ill-conditioning of the geometric matrix associated with the least squares procedure [24]. The positivity property is an important ingredient of any industry standard flow solver, because it is a direct measure of the robustness the solver offers. While the limiters [26] render the required robustness to inviscid flow solvers, enforcement of positivity is a difficult proposition for the viscous flux discretization. Here we simply introduce the methodology we have used for the viscous discretization, showing both robustness and cost effectiveness. Of course,



a: A 2-D Cluster b: Illustration for split fluxes

Fig. 2. Cluster for LSFD-U.

this procedure is by no means unique and different possibilities towards viscous discretization and their positivity as applied to the model Laplace equation are presented in Appendix A. We defer discussions on ill-conditioning of the least squares geometric matrix, in the context of the spatial discretization associated with a turbulent wall boundary layer to Section 4.

The discretization procedure for viscous terms basically involves expanding the viscous flux derivatives into its components involving, both first and second derivatives of velocities and temperature. For example, the viscous flux derivatives appearing in the x momentum equation read:

$$\frac{\partial \tau_{xx}}{\partial x}+\frac{\partial \tau_{xy}}{\partial y}=\frac{1}{Re_\infty}\left[\frac{\partial(\mu+\mu_t)}{\partial x}\left(2\frac{\partial u}{\partial x}-\frac{2}{3}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right)+(\mu+\mu_t)\left(2\frac{\partial^2 u}{\partial x^2}-\frac{2}{3}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 v}{\partial x\partial y}\right)\right)+\frac{\partial(\mu+\mu_t)}{\partial y}\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)+(\mu+\mu_t)\left(\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 v}{\partial y\partial x}\right)\right]. \quad (8)$$

The first and second solution derivatives are obtained using a quadratic least squares procedure. The derivatives of temperature required in the discretization of energy equation are derived from the derivatives of pressure and density using the equation of state. Similarly, the derivatives of fluid viscosity and thermal conductivity are recovered from the derivatives of temperature, using the Sutherland's formula [23].

2.5. Accuracy of the procedure

Based on the discussions presented in Sections 2 and 5 of Ref. [9], it is clear that the proposed discretization procedures, both inviscid and viscous, are first order accurate on a general point distribution. It should also be remarked that the proposed procedures are second order accurate on a regular point distribution and the accuracy degenerates to first order on an irregular general point distribution. This is similar to the order of accuracy exhibited by unstructured grid based finite volume procedures. Though these methodologies formally exhibit first order accuracy on irregular grids (point distribution in the present case), the associated global errors fall at a rate close to two, with progressive grid refinement [42]. Therefore the first order accuracy of these methodologies, both the proposed meshless solver and the unstructured grid based finite volume method, should not be construed as a limitation in obtaining accurate solutions (also evident from the results produced in the present work). Apart from this, as discussed in Ref. [9], it is possible to systematically build schemes of required order of accuracy, within meshless framework, though in the present studies we restrict ourselves to first order accurate schemes suitable for practical computations.

3. Point generation

Quite often, while dealing with the topic of obtaining point distribution for meshless solvers, the authors remark that it may be obtained using any one of the existing grid generators; structured, unstructured or cartesian[1] [27]. We consider that a robust point generation strategy is very important to realize the full potential of the meshless solvers. In our view, the cartesian meshes provide the most natural way to obtain the required point distribution. We had proposed a grid stitching strategy for generating cartesian like grids, which completely avoids the appearance of small cut cells, in the context of finite volume methodology [5]. A point distribution obtained from such a procedure has also been used within the framework of meshless solvers [20]. While this strategy can be successfully employed for inviscid and laminar flow computations, its extension to turbulent flows is not straightforward [20,24]. Therefore, the present work deals with the use of a generic point distribution strategy, inspired by the developments in the cartesian meshes, for computing turbulent flows. Considering the inevitability of the hybrid cartesian mesh for RANS computations, the point distribution is obtained from such a grid.

The details of the hybrid cartesian mesh generation may be had from Refs. [2,3,24]. The procedure simply involves:

1. Generation of highly stretched structured grid (around each of the elements in case of multi-element airfoils) using any of the standard hyperbolic mesh generation tools [28], as depicted in Fig. 3a for an NLR two element configuration.
2. Deletion of the overlap region, leaving us with a structured grid front as depicted in Fig. 3b.
3. Filling the computational domain with an automatic cartesian gridding strategy and deleting the cartesian grid points falling within the structured grid front; this is shown in Fig. 3c and d.
4. Deletion of cartesian grid points falling close to the structured grid front based on a simple distance based criterion; this operation may be considered analogous to the small cut cell treatment associated with cartesian mesh based finite volume computations, although it is considerably simpler.

3.1. Point sorting

It is important to note that, a vast majority of points, falling both in the structured and cartesian grid blocks exhibit an inherent structure. One of the key aspects of the present algorithm is to exploit this local structure in deriving appropriate discretization schemes, particularly with an objective of realizing a robust solution procedure. Therefore, the point sorting algorithm presented in this section becomes an important component of the point generation strategy. The points generated using the hybrid cartesian mesh generation strategy presented in the previous section, are sorted as follows: (a) structured type; (b) cartesian type; (c) hanging type; (d) general type; and (e) boundary type.

3.1.1. Structured type

Points identifiable with a south (S), east (E), north (N) and west (W) neighbourhood as depicted in Fig. 4 fall under this category. The interior points in the structured grid block naturally belong to this type. The discretization procedures associated with the points belonging to this type (to be discussed in Section 4) exhibit additional robustness. Therefore, we also include the points on the structured grid front and certain points on the cartesian grid front into the structured type. It is important to note that for these points falling on the grid fronts, one or two points in the structured neighbourhood may be missing. The slots for the missing neighbours are filled with appropriate points belonging to the other front. A pictorial depiction of the choice of the missing neighbours, in the case of an example involving a point falling on the cartesian grid front and a corner point (two of whose neighbours are missing) of a structured grid front is presented in Fig. 5. In the figure, for the point c on the cartesian grid front, the northern neighbour N is missing. Point i on the structured grid front is used to fill this missing neighbour's slot, because, the angle ψ subtended by the ray $\vec{ci}$ with $\vec{Sc}$ is less than a predefined value θ ($\theta = 30^\circ$ in the present computation). Similar procedure is employed for other points lying on both structured and cartesian grid fronts. Interestingly, given the fact that the cartesian grid is generated to resolve the length scales associated with the structured grid front, the missing

[1] This is quite contrary to the common perception in the CFD community that the cloud of points required for the meshless solvers are generated using some random numbers!

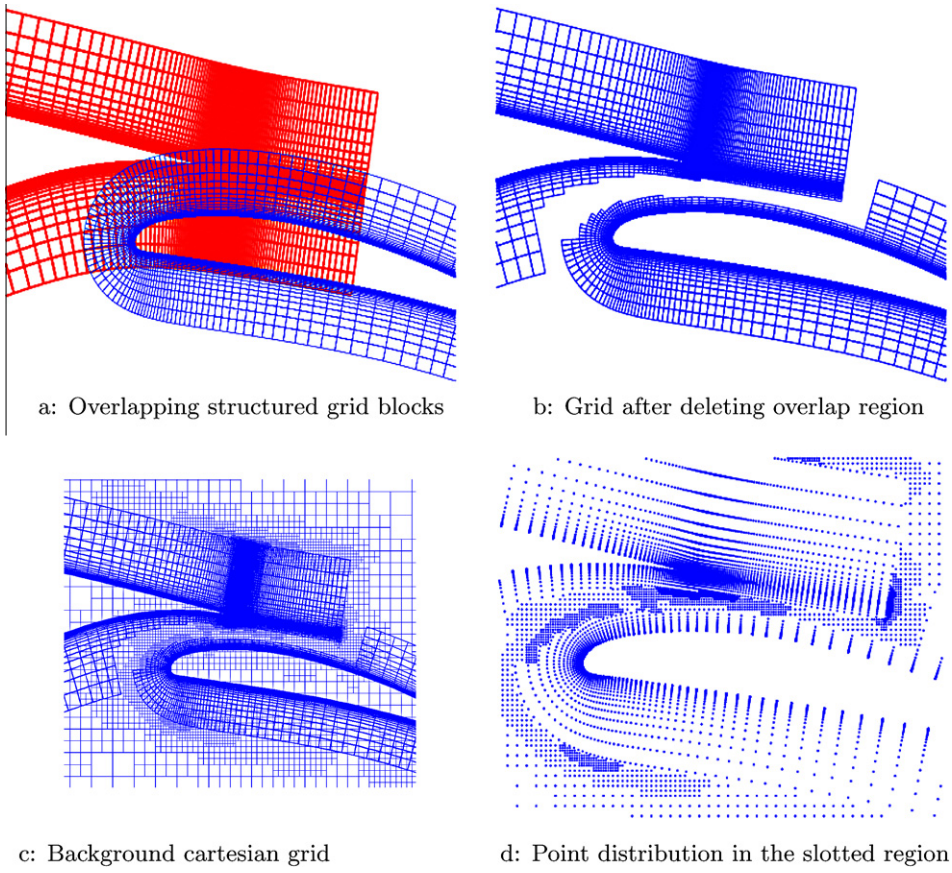


Fig. 3. Hybrid cartesian point generation.

neighbours for all the points belonging to the structured grid front can be had from the points belonging to the cartesian grid front; the vice versa may not be true. The points on the cartesian grid front not categorized under the structured type distribution, may be included either in the hanging type or general type, depending on the connectivity they exhibit.

3.1.2. Cartesian type

The cartesian point identifiable with S, E, N, W neighbourhood belongs to the cartesian type as depicted in Fig. 6 (point c). These four points are said to constitute the primary neighbourhood. Along with this neighbourhood, a secondary support set required for solution update is also identified. For this purpose, all the cells sharing the cartesian point are found. The points constituting these cells excluding the S, E, N, W points form the secondary neighbourhood. The cartesian type points can be present anywhere in the cartesian grid block. For the cartesian type points present on the cartesian grid front, two points from the structured grid front are also included in the secondary support set, based on the proximity criterion.

3.1.3. Hanging type

A point falling in the cartesian grid block (either in the interior or on the front) is categorized as Hanging type [2] if one of its S, E, N, W neighbours is missing. These points are further classified as hanging type 1, if the north (N) or south (S) neighbour is missing and hanging type 2, if the east (E) or west (W) neighbour is missing. Akin

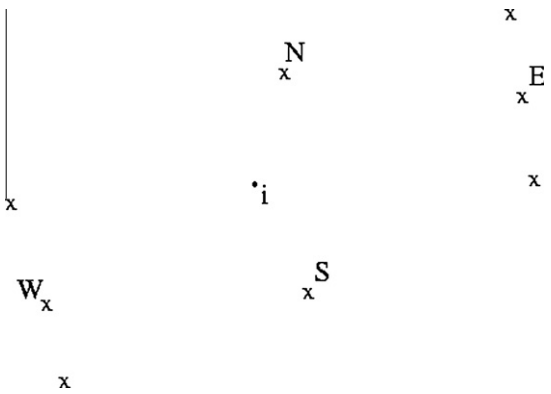


Fig. 4. Support for structured type point.

to the cartesian type points, a secondary neighbourhood is collected for the hanging type points also. For the hanging type points falling on the cartesian grid front, in addition to the secondary neighbours from the cartesian grid block, two nodes from the structured grid front are also included into the secondary neighbourhood, based on the proximity criterion.

3.1.4. General type

A point falling in the cartesian grid front (point *c* in Fig. 7), not classified either as a structured or as a hanging type point, is referred to as a general type point. A generic neighbourhood which includes both the primary and the secondary neighbours (as in the case of hanging type points on the cartesian grid front) is collected for the general type points.

[2] Note the difference in the definition of the hanging type points from that of hanging nodes commonly used in finite volume literature; hanging nodes in finite volumes appear on a face if there is level difference in the cells sharing that face.

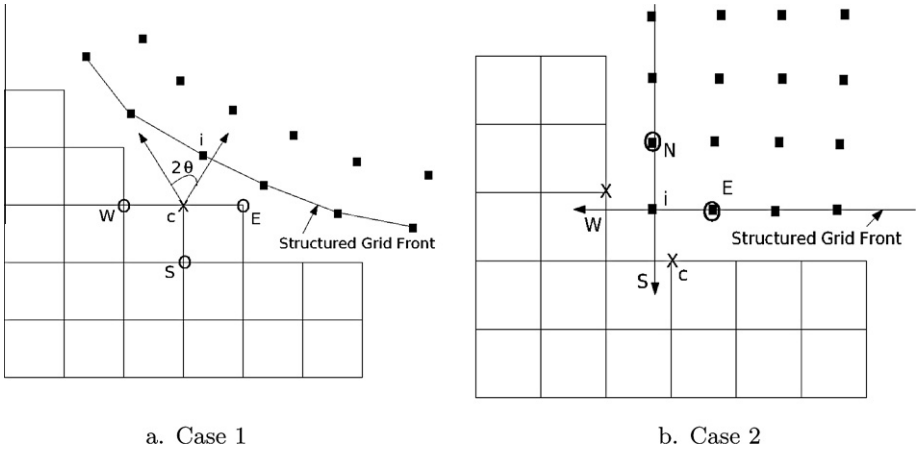


Fig. 5. Support for cartesian and structured grid front points.

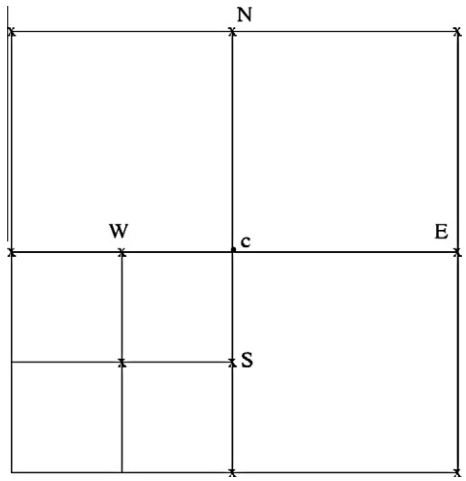


Fig. 6. Support for cartesian type point.

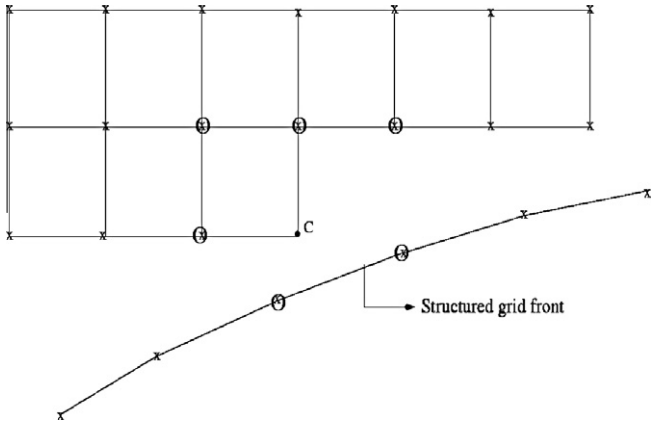


Fig. 7. General type point.

3.1.5. Boundary type

Points falling on the wall boundary belonging to the structured grid block and those falling on the farfield boundary belonging to cartesian grid block are classified as boundary type.

4. LSFD-U flow solver

In Section 2, we have presented a generic discretization strategy for both inviscid and viscous flux derivatives based on the method of least squares. In Section 3, we have brought out the inevitability of padding the viscous layer with structured grid comprising of high aspect ratio quadrilateral volumes, for any practical RANS computations. This being the case, any discretization procedure used in conjunction with such a point distribution should exhibit the required robustness, to become a standard for developing any 3D industrial flow solver. In this context, we can see that both the viscous discretization procedures presented in Appendix A do not satisfy the positivity criterion as applied to a Laplacian. In addition to this, in this section, we bring out yet another limitation of any least squares based solver. For this purpose, consider a typical structured grid block shown in Fig. 8, required for a turbulent flow computation for a RAE airfoil. It should be remarked that this grid exhibits an average y^+ of 1.5, for standard computations involving RAE [29]. From Fig. 9a, it can be noticed that the grid involves wall cells of high aspect ratio (as high as 3500 at the mid-chord). The condition number of the geometric matrix associated with the least squares procedure, for the first layer of points present in the above grid, is given in Fig. 9b. It can be noted that the condition numbers are as high as few million for the points present in the boundary layer, indicating that these geometric matrices are near singular. This behavior of the least squares procedure is not surprising, because, in effect we are trying to recover a two dimensional information in terms of gradients of the flow variables, from a geometry which is essentially one dimensional; but this grid is inevitable to resolve the disparate gradients the physics presents in the streamwise and normal directions. This study clearly reveals the inadequacy of the least squares based procedures for handling highly anisotropic point distribution required for resolving the turbulent flow features. In addition, it is worthwhile to take note that use of weighted least squares does not render required robustness to the flow solver, particularly for point distributions required for RANS computations, though the condition numbers

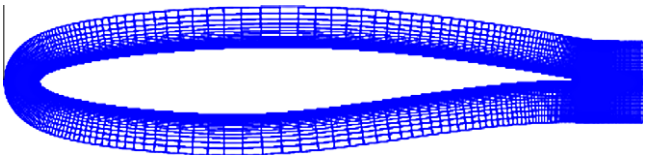
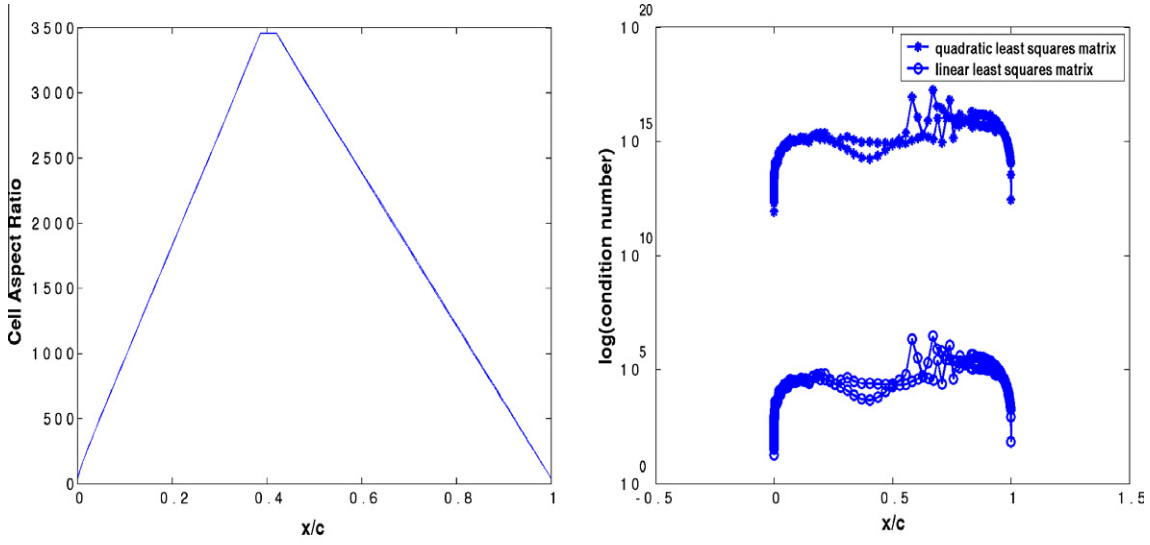


Fig. 8. Structured grid block around RAE Airfoil.



a. Cell aspect ratio distribution b. Condition number distribution

Fig. 9. Cell aspect ratio and condition number for wall cells.

show marked improvement [24]. These observations lead us into looking for discretization options, which exploit the local structure in the grid, while not loosing the generality of the flow solver itself. The classification of the points, presented in the previous section, was done precisely with this objective in mind. In this section, we present discretization procedures to be adopted for each of these point types. Of course, it should be remarked that other than the discretization procedures suggested in this work, for the points lying in the cartesian grid block, there could be few other discretization options, including the one of using a conventional finite volume update. While these options have the potential to be explored further, the present procedure naturally fits in the meshless framework and is proven to be robust and accurate.

In the following discussions, it is important to note that the gradients of the solution variables available at nodes are used for the purpose of solution reconstruction. These gradients along with the Hessians are also used for defining the viscous flux derivatives. For obvious reasons, these gradients are limited after they have been used in the viscous discretization, for the purpose of determining the upwind inviscid fluxes at the fictitious interface.

4.1. Structured type

Most of the structured type points are derived from a body fitted grid. Therefore, they naturally exhibit an alignment with the streamline coordinate. We exploit this feature, by solving the conservation laws on a locally rotated coordinate. With reference to Fig. 10, the ray *iE* represents the ξ coordinate and η refers to the direction normal to it. The governing equations on the rotated coordinates reads,

$$\frac{\partial \widetilde{W}}{\partial t}+\frac{\partial(\tilde{f}+\widetilde{F})}{\partial \xi}+\frac{\partial(\tilde{g}+\widetilde{G})}{\partial \eta}=0 \tag{9}$$

where the vector of conserved variable is $\widetilde{W}=[\rho \quad \rho\tilde{u} \quad \rho\tilde{v} \quad e]^T$. Here, the expression for the flux vectors involves the velocity components in rotated coordinates, $\tilde{u}$ and $\tilde{v}$ and the gradients are in ξ and η directions.

4.1.1. Viscous discretization

As described in Section 2.4, discretization of the viscous flux derivatives involves approximations to both the gradients and

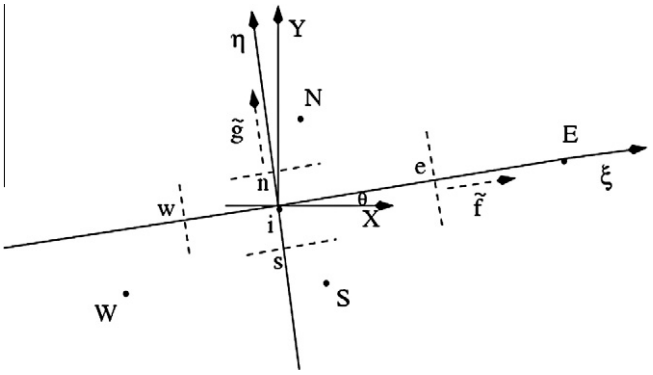


Fig. 10. Structured type point.

Hessians of the primitive variables, which are determined using the Green-Gauss procedure [29,30]. A closed path is defined around the node *i*, as shown in Fig. 11. For an arbitrary variable ϕ, the Green Gauss theorem reads,

$$\int_{\Omega_i}\nabla\phi\, d\Omega=\int_{\Gamma_i}\phi\, d\vec{\Gamma}. \tag{10}$$

The discrete approximation to the above equation is given by,

$$\nabla\phi_i=\frac{1}{\Omega_i}\sum_{k=1}^{4}\phi_k\hat{n}_k\Delta S_k, \tag{11}$$

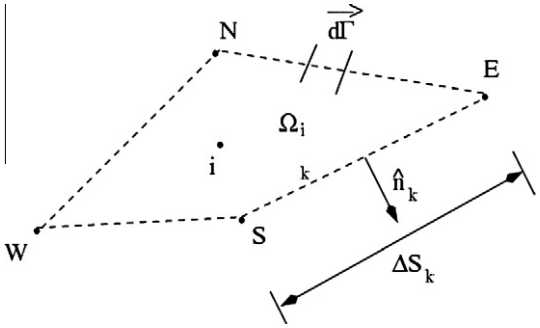


Fig. 11. Closed path around a structured type point.

where $\hat{n}_k$ and ΔS_k represent the unit normal and length of the kth edge falling on the closed path. Using Trapezoidal rule, ϕ_k is obtained as an average of the solution values at the nodes constituting the kth edge. The volume bounded by the closed path Ω_i is defined as,

$$\Omega_i = \sum_{k=1}^{4} x_k n_{x_k} \Delta S_k. \tag{12}$$

The gradients thus obtained are accurate to first order. The second order derivatives of the primitive variables are obtained exactly using the same procedure, by making use of the gradients already available at the nodes. The expression for the second order derivatives read,

$$\phi_{xx_i} = \frac{1}{\Omega_i} \sum_{k=1}^{4} \phi_{x_k} n_{x_k} \Delta S_k, \tag{13}$$

$$\phi^{(1)}_{xy_i} = \frac{1}{\Omega_i} \sum_{k=1}^{4} \phi_{x_k} n_{y_k} \Delta S_k, \tag{14}$$

$$\phi^{(2)}_{xy_i} = \frac{1}{\Omega_i} \sum_{k=1}^{4} \phi_{y_k} n_{x_k} \Delta S_k, \tag{15}$$

$$\phi_{yy_i} = \frac{1}{\Omega_i} \sum_{k=1}^{4} \phi_{y_k} n_{y_k} \Delta S_k \tag{16}$$

and

$$\phi_{xy_i} = \frac{\left(\phi^{(1)}_{xy_i} + \phi^{(2)}_{xy_i}\right)}{2}. \tag{17}$$

These derivatives (Eqs. (13)–(17)) are directly employed in computing the viscous flux derivatives in the (x,y) coordinate system. The flux derivatives on the rotated (ξ,η) coordinate system are obtained using the relation:

$$[\tilde{\mathbf{F}}_{\xi_i} + \tilde{\mathbf{G}}_{\eta_i}] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} [\mathbf{F}_{x_i} + \mathbf{G}_{y_i}]. \tag{18}$$

There are three important observations one can make with regard to this viscous discretization procedure. They are:

1. This viscous discretization procedure as applied to a Laplacian is positive and therefore can be expected to be robust.
2. Discrete Laplacian obtained using this procedure exhibits odd–even decoupling. In our view, this observation in itself may not be of significance for establishing a robust procedure to Navier–Stokes equations. This is because of the strong coupling between the odd–even points, coming from the convective terms. Also, recall that the Laplacian is not an adequate model for describing the viscous flux derivatives in the compressible flow equations; the additional terms appearing in the equations also provide the required coupling. In this work, we have produced accurate solutions even for diffusion dominated flows, for Reynolds numbers as low as 40 (see Section 5).
3. Given the fact that the gradients available at the nodes are accurate only to first order, the Hessians are $O(1)$ inconsistent. Interestingly, some of the successful viscous discretization procedures employed for unstructured mesh based finite volume solvers are also known to be inconsistent [24]. These schemes are known to exhibit what is referred to as supraconvergence [31]; i.e., the global errors exhibit consistency while the local truncation terms are inconsistent. In Appendix B, we empirically establish this behavior for the present discretization procedure, for the case of a linear advection–diffusion equation on a point distribution representative of the one encountered in the boundary layer. It should also be remarked that it is not difficult to correct this inconsistency using a recursive defect correction strategy. We have not employed such a strategy, simply because, the positivity property, rendering the required robustness to the present procedure, may be lost.

4.1.2. Inviscid discretization

The concept of fictitious interface introduced within the framework of LSFD-U [9] comes handy in evolving a simple, yet robust inviscid discretization strategy for structured type points. In the conventional LSFD-U procedure (as described in Section 2.3), the fictitious interface is introduced at the mid-point of the ray joining two nodes. On the contrary, for the structured type points, the fictitious interfaces are introduced on the coordinate lines. The location of the fictitious interface is obtained by projecting the mid-point of the ray connecting a node with its neighbour on to an appropriate coordinate line. In Fig. 10, the fictitious interface marked by n, e, s and w correspond to the neighbours N, E, S and W. The gradients available at the nodes are made use of for obtaining the left and right states at the fictitious interfaces. Using any upwind formula, fluxes along the coordinate directions are determined at the interfaces and the flux derivatives are obtained using simple 1D least squares formula:

$$\tilde{f}_{\xi_i} = \frac{\sum_J \Delta\tilde{f}_J \Delta\xi_J}{\sum_J \Delta\xi_J^2}; \quad J \in [e,w], \tag{19}$$

$$\tilde{g}_{\eta_i} = \frac{\sum_J \Delta\tilde{g}_J \Delta\eta_J}{\sum_J \Delta\eta_J^2}; \quad J \in [n,s]. \tag{20}$$

Since the update involves ξ and η momentum equations, the velocities thus obtained are rotated back to the (x,y) coordinates and stored.

4.2. Cartesian type

For points belonging to cartesian grid block (whether it is of cartesian type or hanging type), whenever the nodes are aligned with the coordinate direction, for reasons of economy, a simple 1D finite difference approximation is employed to determine the derivatives. In that sense, the cartesian type points, characterized by the presence of a complete primary neighbourhood, require most straightforward differencing procedure.

4.2.1. Viscous discretization

With reference to Fig. 12, defining

$$\phi^e_x = \frac{\phi_E - \phi_i}{x_E - x_i}, \quad \phi^w_x = \frac{\phi_W - \phi_i}{x_W - x_i}, \tag{21}$$

and

$$\phi^n_y = \frac{\phi_N - \phi_i}{y_N - y_i}, \quad \phi^s_y = \frac{\phi_S - \phi_i}{y_S - y_i}, \tag{22}$$

we have,

$$\phi_{x_i} = \frac{\delta_w \phi^e_x + \delta_e \phi^w_x}{\delta_e + \delta_w}, \tag{23}$$

$$\phi_{y_i} = \frac{\delta_s \phi^n_y + \delta_n \phi^s_y}{\delta_n + \delta_s}, \tag{24}$$

where $\delta_e = |x_E - x_i|$, $\delta_w = |x_W - x_i|$, $\delta_n = |y_N - y_i|$, $\delta_s = |y_S - y_i|$.

The second derivatives $\phi_{xx,i}$ and $\phi_{yy,i}$ are also obtained using a simple centered approximation:

$$\phi_{xx_i} = \frac{\phi^e_x - \phi^w_x}{x_e - x_w}, \tag{25}$$

and

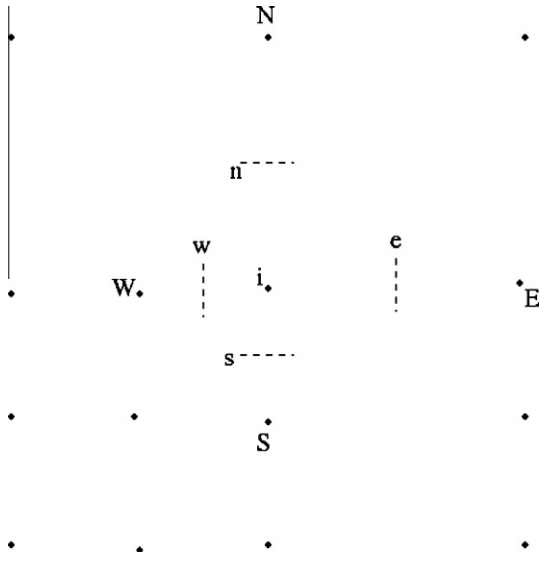


Fig. 12. Cartesian type point.

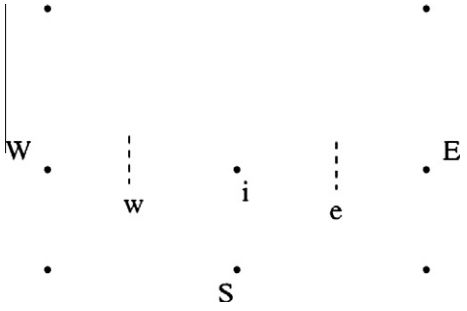


Fig. 13. Hanging type point 1.

$$\phi_{yy_i} = \frac{\phi_y^n - \phi_y^s}{y_n - y_s} \tag{26}$$

where $x_e = \frac{x_E + x_i}{2}$ and so on.

The approximation for the cross derivative ϕ_{xy} is obtained using a modified least squares procedure, including a full stencil of points (comprising of both the primary and secondary neighbours) in the connectivity. For this purpose, treating all the derivatives except the cross derivative ϕ_{xy} as known quantities, a modified difference term can be defined as follows:

$$\widetilde{\Delta\phi_j} = \Delta\phi_j - \left(\phi_{x_i}\Delta x_j + \phi_{y_i}\Delta y_j + \phi_{xx_i}\frac{\Delta x_j^2}{2!} + \phi_{yy_i}\frac{\Delta y_j^2}{2!}\right). \tag{27}$$

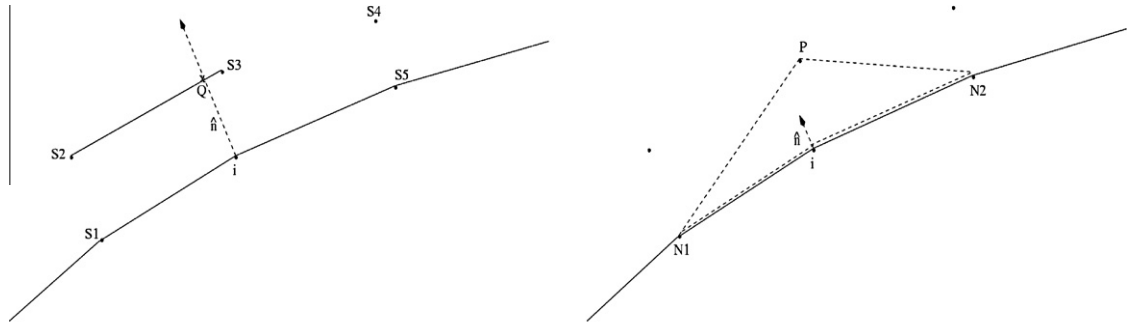
The above equation defines an over determined system and the use of method of least squares results in the following approximation for the cross derivative:

$$\phi_{xy_i} = \frac{\sum_j \widetilde{\Delta\phi_j}\Delta x_j \Delta y_j}{\sum_j (\Delta x_j \Delta y_j)^2}. \tag{28}$$

For a simple cartesian type point, not having hanging nodes in its neighbourhood, the determination of the cross derivative ϕ_{xy}, effectively will involve only the diagonal nodes. It should be remarked that for the cartesian type points, gradients are determined to second order accuracy and the Hessians are determined to first order accuracy. These derivatives are directly used for approximating the viscous flux derivatives.

4.2.2. Inviscid discretization

The discretization of the inviscid flux derivatives is exactly similar to the way it is done for structured type points. An added simplification is that the local coordinates already correspond to (x,y) coordinates and there is no need for any coordinate rotation.



a. Interpolation and projection b. One-sided closed path

Fig. 14. Wall boundary update.

Table 1
Test cases: details of freestream conditions and point distribution.

Case	Geometry	Flow conditions M_∞, α, Re_∞	Number of points	Number of wall points	General type points (%)	Reference data
1	Circular cylinder	0.1, 0.0, 40	8347	160	1.3	Tritton [36], Ratnesh et al. [37]
2	NACA 0012 Biplane	0.8, 10°, 500	19974	198 + 198	0.9	Jawahar and Hemanth [30]
3	RAE 2822	0.676, 1.92°, 5.7×10^6	26427	295	0.5	Cook et al. [38]
4	RAE 2822	0.73, 2.79°, 6.5×10^6	26427	295	0.5	Cook et al. [38]
5A and 5B	NLR 7301	0.185, 6° & 13.1°, 2.51×10^6	34801	300 + 200	1.3	Berg [39]
6	NACA 0012	0.5, 3°, 5000	11669	200	6.0	Venkatakrishnan [40]

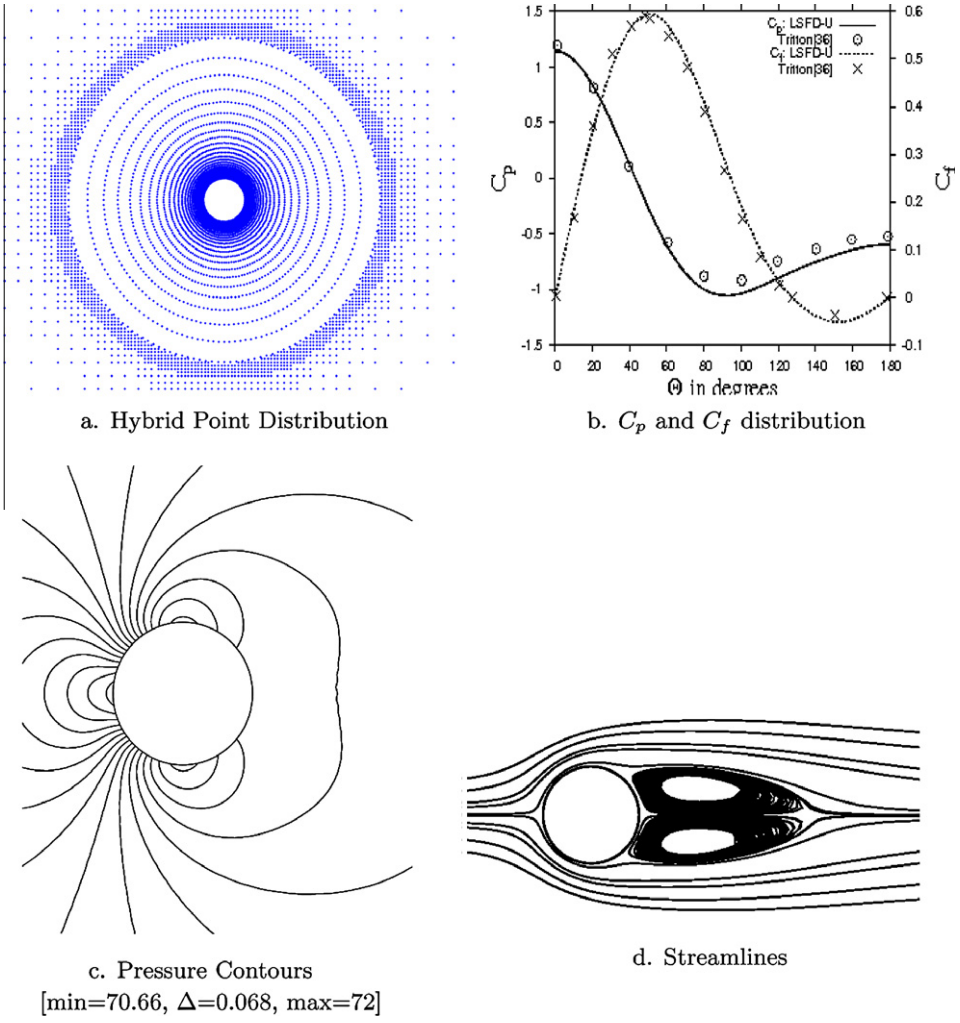


Fig. 15. Case 1, laminar flow: circular cylinder, $Re_\infty = 40$, $M_\infty = 0.1$.

Table 2
Case 1, laminar flow over circular cylinder: wake bubble length, angle of separation and Drag coefficients.

Solution	Wake bubble length, b	Angle of separation, θ_s (°)	C_D
LSFD-U	2.31	127.4	1.63
Refs. [36,37]	2.18–2.35	125.8–127.3	1.48–1.62

4.3. Hanging type

The hanging type points are characterized by the fact that one of the primary neighbours is missing. Therefore, simple 1D approximation to certain derivatives is possible in at least one of the coordinate directions. Here we explain the procedure for hanging type 1 points and the extension of this idea to hanging type 2 points is trivial.

4.3.1. Viscous discretization

With reference to Fig. 13, for a hanging type point, a complete set of primary neighbours in the x coordinate direction is available. Therefore, both the derivatives ϕ_{x_i} and ϕ_{xx_i} are determined using the same procedure used for cartesian type points. The other derivatives are obtained using a modified least squares procedure. For this purpose, we define a modified solution difference, given by,

$$\widetilde{\Delta\phi_j} = \Delta\phi_j - \phi_{x_i}\Delta x_j - \phi_{xx_i}\frac{\Delta x_j^2}{2}. \tag{29}$$

The system defined by the above equation is solved using a least squares procedure. This results in,

$$\mathbf{Ax} = \mathbf{B}, \tag{30}$$

where

$$\mathbf{A} = \begin{bmatrix} \sum_j \Delta y_j^2 & \sum_j \Delta x_j \Delta y_j^2 & \sum_j \frac{\Delta y_j^3}{2} \\ \sum_j \Delta x_j \Delta y_j^2 & \sum_j \Delta x_j^2 \Delta y_j^2 & \sum_j \frac{\Delta x_j \Delta y_j^3}{2} \\ \sum_j \frac{\Delta y_j^3}{2} & \sum_j \frac{\Delta x_j \Delta y_j^3}{2} & \sum_j \frac{\Delta y_j^4}{4} \end{bmatrix} \tag{31}$$

$$\mathbf{x} = [\phi_{y_i} \quad \phi_{xy_i} \quad \phi_{yy_i}]^T, \tag{32}$$

$$\mathbf{B} = \begin{bmatrix} \sum_j \Delta\tilde{\phi}_j \Delta y_j \\ \sum_j \Delta\tilde{\phi}_j \Delta x_j \Delta y_j \\ \sum_j \frac{\Delta\tilde{\phi}_j \Delta y_j^2}{2} \end{bmatrix}, \tag{33}$$

The gradients thus obtained are accurate to second order, while the Hessians are first order accurate.

4.3.2. Inviscid discretization

While a simple 1D least squares approximation is employed for the flux derivatives f_{x_i}, the general procedure discussed in Section 2.3 is used for approximating g_{y_i}. A connectivity which includes both the primary and secondary neighbours is used in the general procedure for determining g_{y_i}.

4.4. General type

For such a point, a quadratic least squares procedure described in Section 2.4 is used for approximating the gradients and Hessians of the primitive variables. For the inviscid flux derivatives, the LSFD-U procedure described in Section 2.3 is made use of. Only a small percentage of points falling on the cartesian grid front are treated as general type points.

4.5. Boundary type

For the class of numerical test cases considered in this work, wall and farfield boundaries are present. Riemann invariant boundary condition is used at the farfield points and strong boundary condition is imposed at the viscous wall points and it is explained below in some detail.

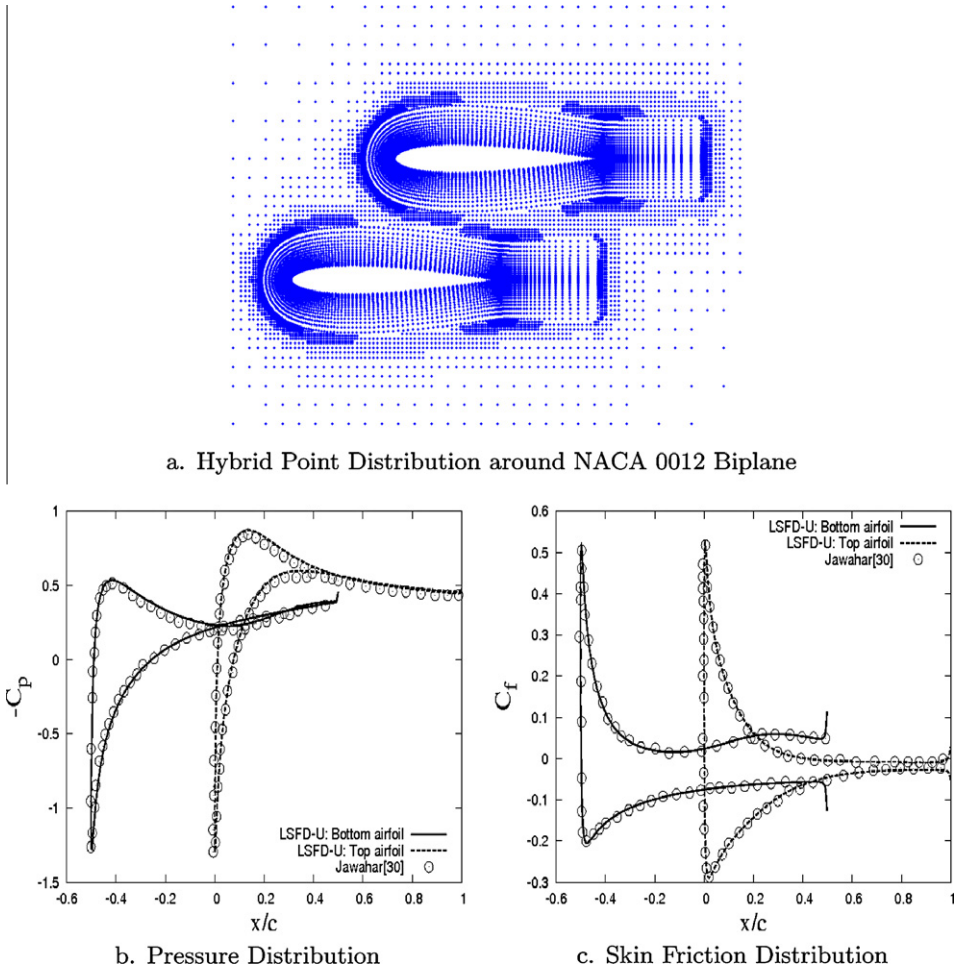
4.5.1. Wall boundary

The no slip boundary condition is imposed by setting the values of velocity components to be zero. From the boundary layer assumption, pressure is recovered from the normal gradient condition $\frac{\partial p}{\partial n} = 0$. Assuming adiabatic wall condition, temperature is recovered from the no normal heat flux condition $\frac{\partial T}{\partial n} = 0$. While the least squares based procedure is generally used for numerically satisfying the condition on normal gradients [9,32], it exhibits the problem of ill-conditioning because of the large anisotropy present in the support points for the wall boundary nodes typically encountered in turbulent grids. Therefore, exploiting the directionality present in the neighbourhood, we have implemented an interpolation and projection strategy. Referring to Fig. 14a, for a wall point i, the line joining two successive support points (here, $S2$ and $S3$) which is intersected by the local normal direction $\hat{n}$ at the wall is identified. The solution values available at the points $S2$ and $S3$ are linearly interpolated to the point of intersection Q and the pressure and temperature values at the wall boundary point are simply obtained as follows:

$$p_i = p_Q \quad T_i = T_Q.$$

For the convex corner points such as trailing edge of an airfoil, due to the difficulty in defining a unique normal to the wall at these points, solution values are interpolated from the neighbouring wall boundary points [24].

The gradient information at the wall boundary point is required for computing wall shear stress and for solution reconstruction. The gradients are computed using Green-Gauss procedure described in Section 4.1.1 by constructing a one-sided closed path as shown in Fig. 14b, where P is the support point which makes least angle with the local normal direction $\hat{n}$.

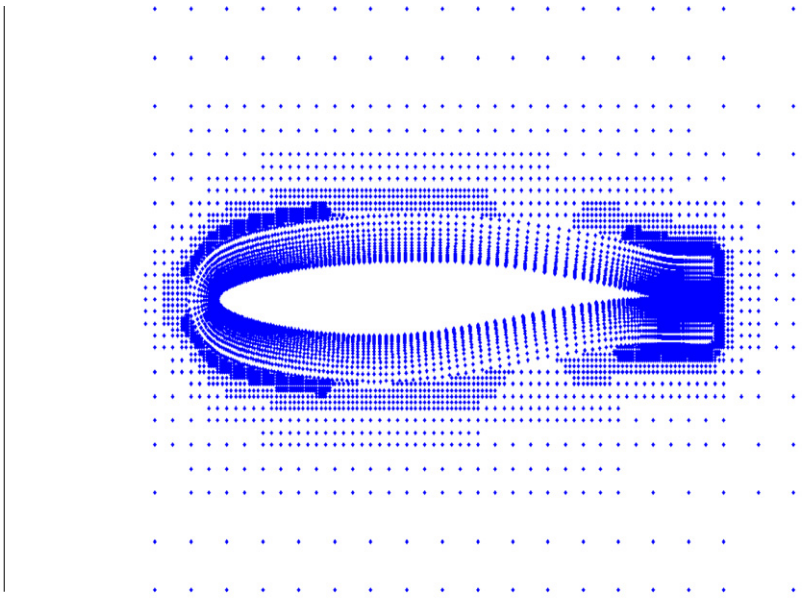


a. Hybrid Point Distribution around NACA 0012 Biplane

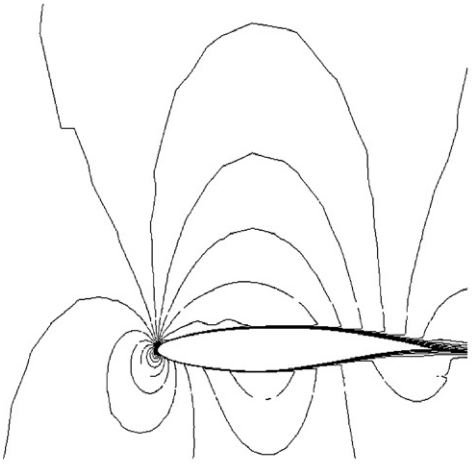
b. Pressure Distribution

c. Skin Friction Distribution

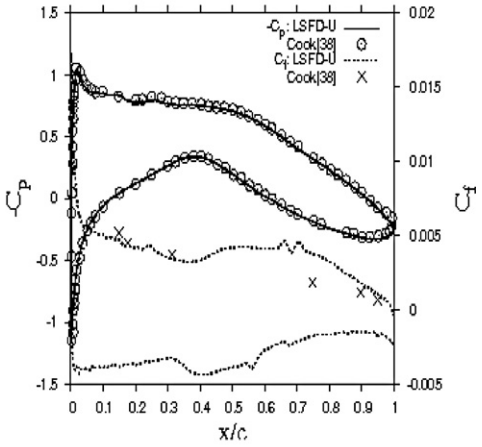
Fig. 16. Case 2, laminar flow: NACA 0012, $M_\infty = 0.8$, $Re_\infty = 500$, $\alpha = 10^\circ$.



a. Hybrid Point Distribution around RAE 2822 Airfoil



b. Mach number contours [min=0.0,Δ=0.05,max=1.05]



c. C_p and C_f distribution

Fig. 17. Case 3, turbulent flow: RAE 2822, $M_\infty = 0.676$, $Re_\infty = 5.7 \times 10^6$, $\alpha = 1.92°$.

5. Results and discussion

The efficacy of the meshless solver LSFD-U along with the hybrid point distribution is demonstrated by solving standard turbulent flow problems. In all the computations presented, Roe scheme [34] is used for computing inviscid fluxes. The solution monotonicity is enforced using Venkatakrishnan's limiter [26]. Baldwin–Lomax turbulence model [35] is employed for computing eddy viscosity. The SGS implicit relaxation procedure akin to the one developed in Ref. [33] is used for accelerating the convergence to steady state. Details of the free stream conditions and point distribution employed are shown in Table 1.

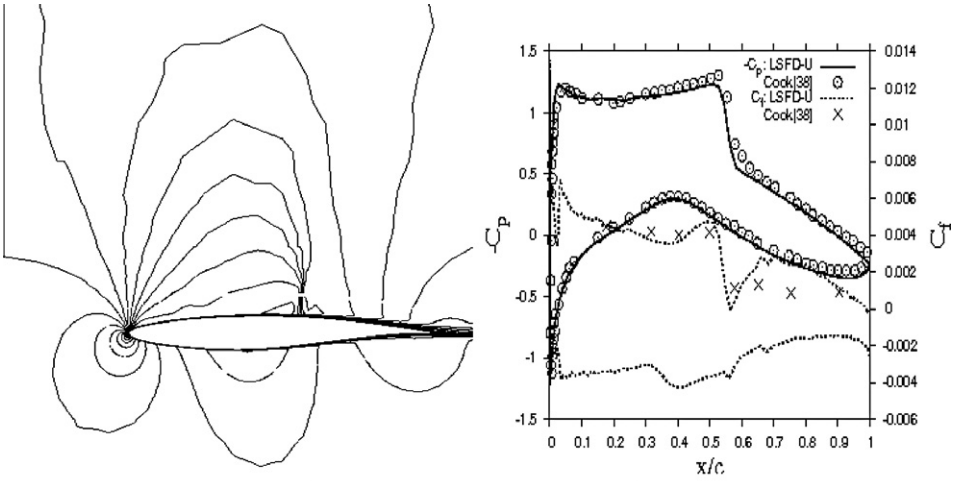
Test cases have been chosen with the objective of either validating the discretization strategy discussed in Section 4 or establishing the efficacy of the meshless solvers in solving flow past complex configurations. The results have been presented for a wide range of Reynolds numbers, from $Re_\infty = 40$ to turbulent flows. The computations have been made using hybrid cartesian point distributions generated around the body of interest. In these point distributions, the structured grid block consists of 30–35 layers of points. For turbulent simulations, the normal placement of the first point off the wall is 1×10^{-5} times the chord of the airfoil resulting in an average y^+ value of 1.5. Farfield is situated at 20 chords away from the body. The solution convergence to steady state is declared after seven decades and six decades of residue fall for laminar and turbulent flow computations respectively.

5.1. Case 1: laminar flow past circular cylinder

This test case involves simulating laminar flow past circular cylinder at $Re_\infty = 40$ based on the diameter of the cylinder. This is an important test case particularly considering the comment 2 on

Table 3
Case 3, subsonic turbulent flow over RAE 2822 Airfoil: lift and drag coefficients.

Solution	C_L	C_D
LSFD-U	0.5545	0.0102
HIFUN-2D [29]	0.5622	0.0125



a. Mach number contours [min=0.0,Δ=0.065,max=1.28] b. C_p and C_f distribution

Fig. 18. Case 4, turbulent flow: RAE 2822, $M_\infty = 0.73$, $Re_\infty = 6.5 \times 10^6$, $\alpha = 2.79°$.

Table 4
Case 4, transonic turbulent flow over RAE 2822 Airfoil: lift and drag coefficients.

Solution	C_L	C_D
LSFD-U	0.8027	0.0175
Experiment [38]	0.8030	0.0168

odd–even decoupling we had made in Section 4.1.1, pertaining to viscous discretization of structured type points.

Zoomed view of the point distribution close to the body is shown in Fig. 15a. Computed pressure and friction coefficient distribution on the upper surface of the cylinder are compared with the experimental values [36] in Fig. 15b. In this figure, $\theta = 0°$ corresponds to the front stagnation point. An excellent agreement with the experiments [36] is found. Iso-pressure contours are depicted in Fig. 15c. The significance of this test case stems from the presence of wake bubble behind the cylinder. This feature is well captured as indicated by the streamline plot in Fig. 15d. A comparison of the predicted values of the wake bubble length b (measured from center of the cylinder), angle of separation θ_s and drag coefficient with the data reported in the literature [36,37] is presented in Table 2. The good agreement indicates that the proposed viscous discretization strategy can accurately predict the strong viscous flow features.

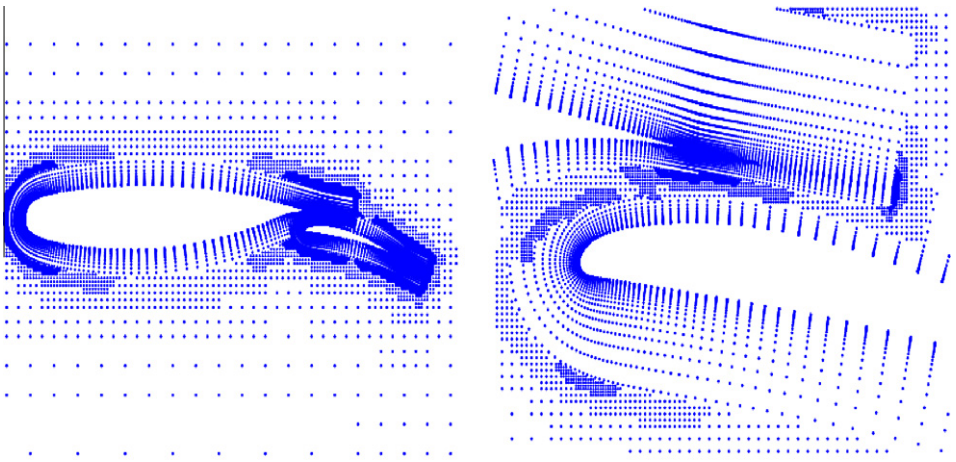
5.2. Case 2: laminar flow past NACA 0012 staggered biplane configuration

This test case involves simulating laminar flow past NACA 0012 staggered biplane at $Re_\infty = 500$ based on the chord length. The two airfoils are staggered by half a chord length in pitchwise and chordwise directions. Zoomed view of the point distribution is shown in Fig. 16a.

The computed pressure and friction coefficient distributions are in good agreement with the profiles reported in other numerical work [30] and the comparison is shown in Fig. 16b and c respectively. These results indicate the efficacy of the LSFD-U procedure in computing flow past multi-component geometries.

5.3. Case 3: subsonic turbulent flow past RAE 2822 Airfoil

This test case involves simulating turbulent flow past RAE 2822 Airfoil at $Re_\infty = 5.7 \times 10^6$ based on the chord, $M_\infty = 0.676$ and the



a: Overall view b: Zoomed view of slot region

Fig. 19. Hybrid point distribution around NLR 7301 Airfoil.

angle of incidence is 1.92°. Zoomed view of the point distribution is shown in Fig. 17a.

Fig. 17b displays the Mach number contours for this test case. Computed pressure distribution and skin friction distributions are compared with experimental values [38] in Fig. 17c. Again, the agreement with the experimental values is good. The computed lift and drag coefficients are presented in Table 3. The predicted aerodynamic coefficients are in good agreement with the values obtained using finite volume solver, HIFUN-2D employing a structured grid as in Ref. [29].

5.4. Case 4: transonic turbulent flow past RAE 2822 Airfoil

All the test cases presented so far involve only continuous subsonic flows. Numerical methodology employed for such flows should necessarily satisfy consistency and stability; which is what is the case for the proposed discretization strategies. These remarks are specifically made in the context of conservative property a numerical procedure should satisfy in order to capture solution discontinuities. It is well known that it is difficult to prove the conservation of generalized finite difference procedures (such as LSFD-U), though there is ample empirical evidence to show that these schemes are capable of capturing discontinuities of right strength at right location. In fact in our earlier work [9], we have systematically established that the LSFD-U procedure can predict accurate results even in case of unsteady problems involving moving shocks on a random point distribution. In furtherance to these observations, we present here a transonic test case involving discontinuous solution.

This test case corresponds to simulating transonic turbulent flow over RAE 2822 airfoil with freestream Mach number 0.73, Reynolds number 6.5×10^6 and angle of incidence of 2.79°. The point distribution is same as the one used in Case 3. A shock is formed on the upper surface of the airfoil and can be seen from the Mach number contours displayed in Fig. 18a. Computed pressure and friction coefficient distributions are shown in Fig. 18b. The surface coefficient values compare well with the experimental data [38] and the shock is captured with correct jump and at right location. Also, the computed aerodynamic coefficients are in good agreement with experimental values [38] and are presented in Table 4. It is evident from these results that the proposed viscous solution methodology can accurately predict the flows with shocks also.

5.5. Case 5: turbulent flow past NLR 7301 Airfoil

This test case investigates turbulent flow past NLR 7301 supercritical airfoil with a slotted trailing edge flap. The chord of the flap

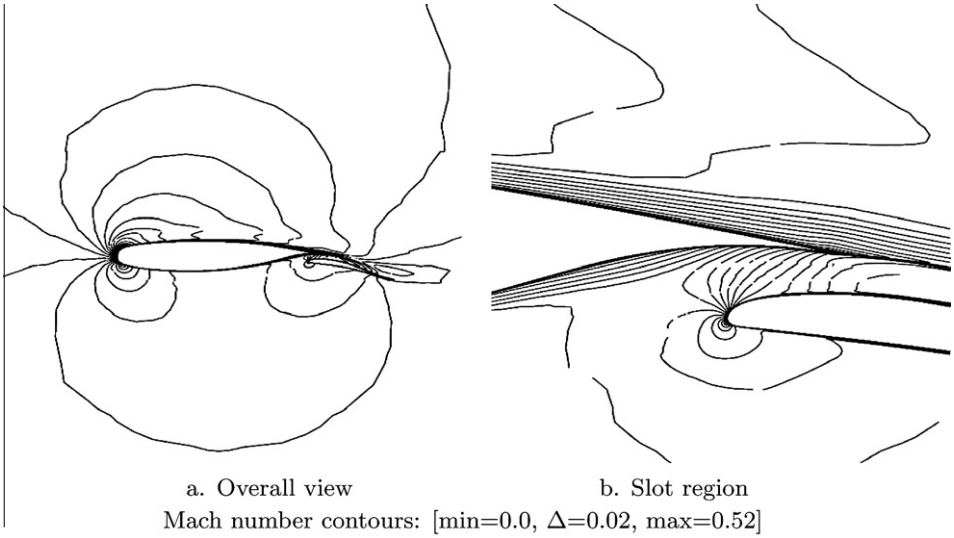


Fig. 20. Case 5A, turbulent flow: NLR 7301, $M_\infty = 0.185$, $Re_\infty = 2.51 \times 10^6$, $\alpha = 6°$.

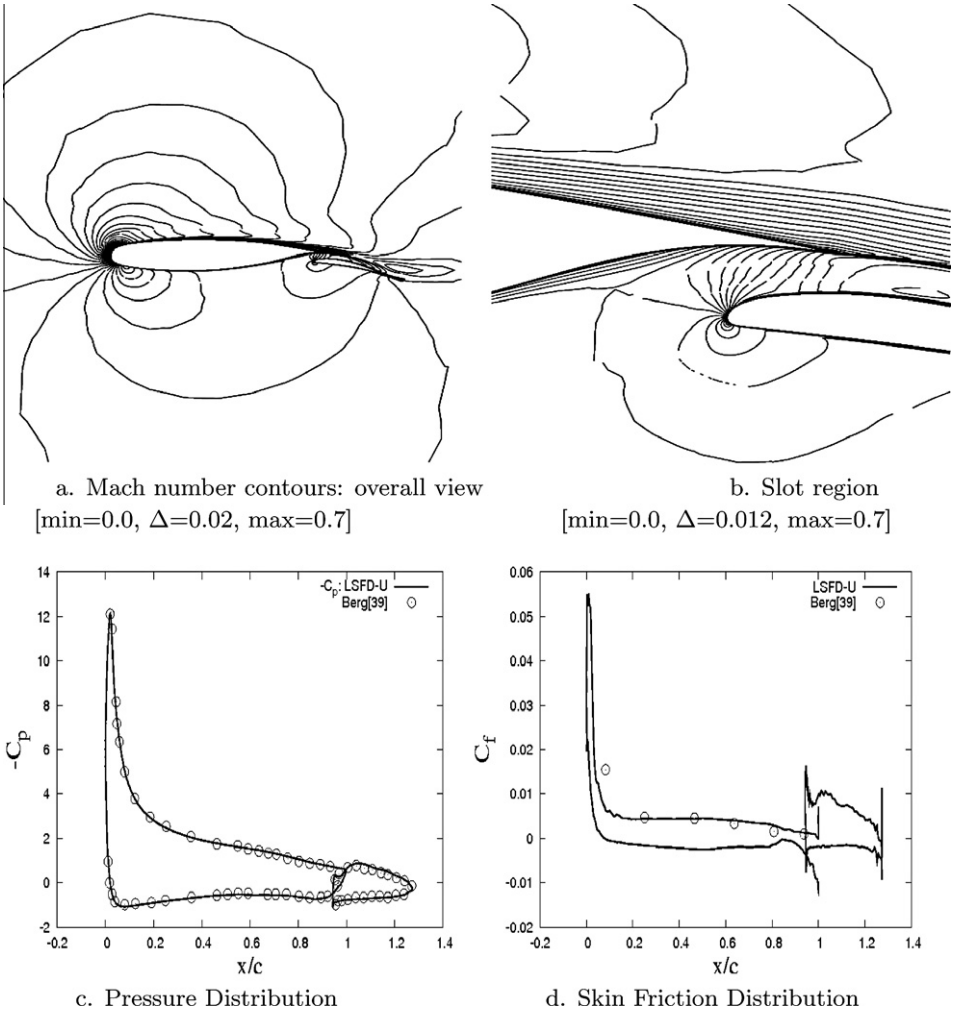


Fig. 21. Case 5B, turbulent flow: NLR 7301, $M_\infty = 0.185$, $Re_\infty = 2.51 \times 10^6$, $\alpha = 13.1^\circ$.

is 32% of the main airfoil chord c, and the flap is deflected at 20°. The overlap distance, i.e., the difference between the x-coordinate of the flap leading edge and x-coordinate of the main airfoil trailing edge is set to be 5.3%. This configuration is similar to a takeoff flap setting and experimentally studied in Ref. [39]. This test case is chosen to demonstrate the efficacy of hybrid cartesian point distribution in handling more complex geometries. The freestream Mach number is 0.185 and the Reynolds number is 2.51×10^6 based on the chord of the main airfoil. Two test cases have been considered corresponding to the angle of incidences, $\alpha = 6^\circ$ and 13.1°.

Fig. 19a depicts the hybrid cartesian point distribution. Close view of the point distribution in the slotted region is presented in Fig. 19b. The computed iso-Mach number contours around complete configuration is shown in Fig. 20a. Its close view in the slotted region is shown in Fig. 20b. The computed pressure distribution is plotted along with experimental values [39], in Fig. 20c and good agreement is observed. As shown in Fig. 20d, the computed skin friction coefficient show an excellent agreement with experimental values [39].

For the flow with an angle of attack of 13.1°, the computed iso-Mach number contours around complete configuration is shown in Fig. 21a. Its close view in the slotted region is shown in Fig. 21b. In Fig. 21c, the computed pressure distribution shows an excellent agreement with experimental values [39]. The computed skin friction coefficient distribution also agrees well with the experiment data [39] and the comparison is shown in Fig. 21d.

5.6. Computational efficiency of LSFD-U

One of the important parameters, for any solver proposed to be used in routine industrial computations, is computational efficiency. This aspect is understood from the implication of using the solver in a design cycle (i.e. from the time we have the surface definition to the time we have post-processed aerodynamic data). In this sense, the present LSFD-U solver can be expected to drastically cut the cycle time, simply because the time to generate the required point distribution around complex geometries can be considerably shorter, as against volume generation. In addition the proposed point generation strategy also offers considerable automation (one of the important advantages of a cartesian mesh generation scheme), another desired feature in an industrial computation. Independent of these major advantages the proposed LSFD-U solver enjoys over its finite volume counterpart in a typical design cycle, we demonstrate in this section the solver in itself is as efficient as a finite volume solver of comparable accuracy. This

Table 5
CPU time per node for one time step.

Point type	Computational time per time step per node $\times 10^{-5}$ (seconds)
Cartesian	1.8
General	5.8
Structured	2.3

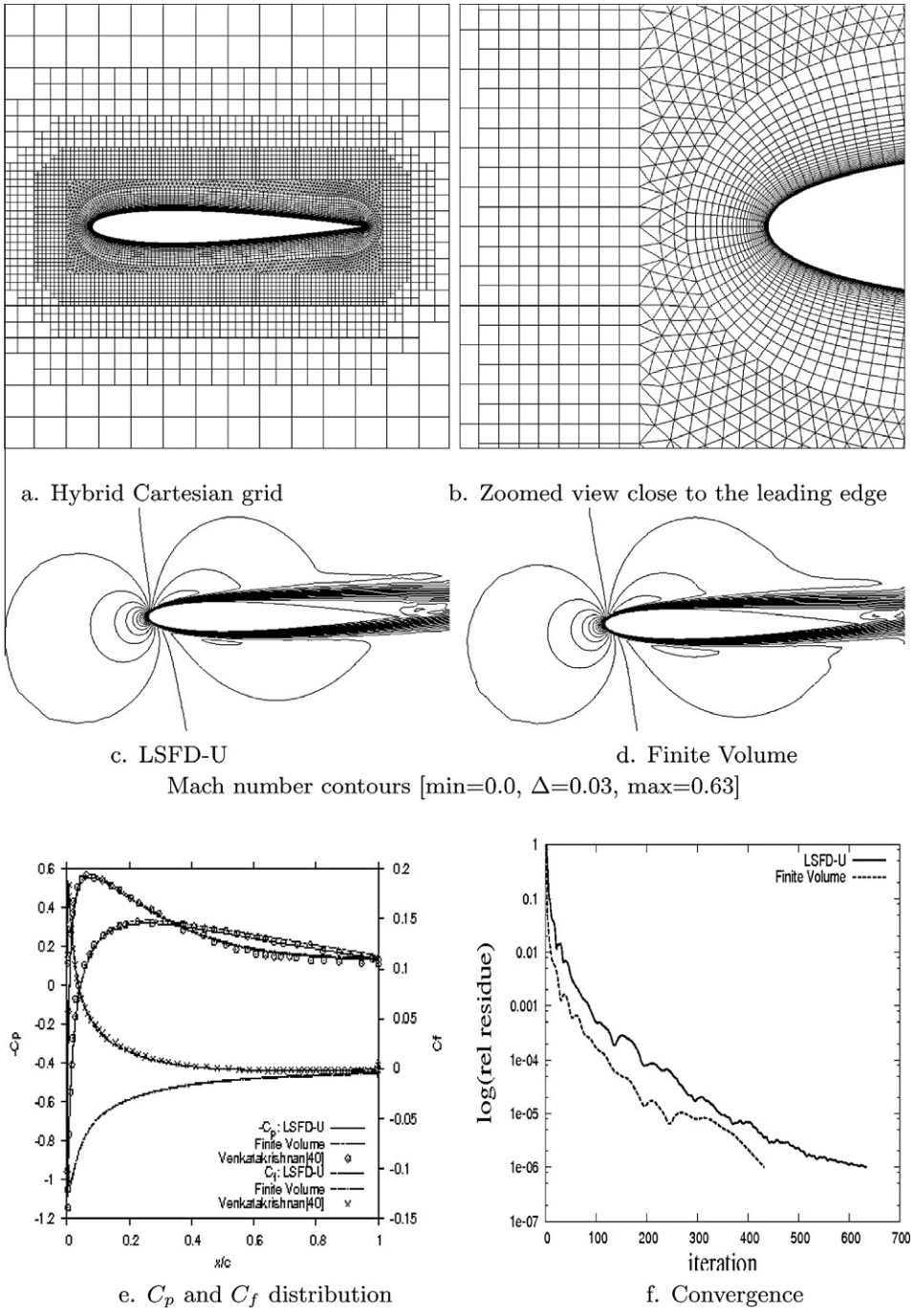


Fig. 22. Case 6, laminar flow: NACA 0012, $M_\infty = 0.5$, $Re_\infty = 5000$, $\alpha = 3°$.

observation is in contrast to the popular perception that the meshless solvers are more expensive as compared to finite volume solvers. This appreciation simply dawns from the fact that we need only one flux computation per edge on the connectivity graph in both the LSFD-U solver and the finite volume solver [9] and therefore there is no reason for the meshless solver to be considerably more expensive than the finite volume solver, as long as they operate on the same connectivity graph.

Even before an actual comparison of the computational efficiency is taken up, it is worthwhile to understand the non-uniformity in the computational effort involved in the nodal update in the proposed viscous LSFD-U solver. For this purpose we consider a simple cartesian domain and update the nodes by considering the nodes as (1) cartesian type points (2) General type points employing a nine node molecule for solution update (3) Structured type points. The computational effort for each of these types obtained on a Linux platform with Dual Core AMD opteron processor with 2600 MHz are presented in Table 5. From the table it is clear that the general type point update is considerably more expensive as compared to both the cartesian and structured types. In light of this observation it is worthwhile to appreciate that the general points constitute only a very small percentage of the total number of nodes as presented in Table 1. Therefore in effect the additional computational effort required by the general type points is not of any significant consequence in the solution update for the entire domain.

It is obvious that a comparison of the computational efficiency of the meshless and the finite volume solvers on a realistic grid/point

Table 6
Case 6, laminar flow over NACA 0012 Airfoil: lift and drag coefficients.

Solution	C_L	C_{D_p}	C_{D_f}	$C_{D_{total}}$	x_{sep} (% of chord)
LSFD-U	0.0504	0.0276	0.0325	0.0601	47.5
HIFUN-2D	0.0469	0.0283	0.0337	0.0620	47.0
Venkatakrishnan [40]	0.0464	0.0262	0.0321	0.0583	46.4

Table 7
CPU time per time step for LSFD-U and finite volume solver HIFUN-2D.

Point Type	Computational time per time step in seconds
LSFD-U	0.28
HIFUN-2D [29]	0.25

distribution discussed in Section 3 is just not possible because of the limitations of the finite volume solver.[3] Therefore, for the sake of comparison we have generated a grid, shown in Fig. 22a, with a structured viscous padding around the body, cartesian grids in the far field and the buffer layer of triangles. While such a grid is naturally amenable for a finite volume update, a meshless update would deal with the points in the triangulated domain as belonging to general type. A point distribution thus obtained has considerably large percentage of general points (6% in the present case) as against the proposed point distribution used in other sections. It was not possible to generate a more realistic point distribution for the sake of comparison, because the triangulation procedure on smaller buffer regions failed. In that sense the computational times presented for the meshless solvers (in Table 7) actually over estimate the CPU requirement of such solvers. It should also be remarked that the finite volume results have been generated using an industry standard implicit solver HIFUN [29], while the meshless solver is a research level code, primarily developed with the objective of testing the basic idea and therefore several enhancements for efficiency, such as implicit wall treatment, are still possible.

5.6.1. Case 6: laminar flow past NACA 0012 Airfoil

The performance of LSFD-U solver is evaluated against the finite volume procedure in terms of solution accuracy and computational efficiency by computing a laminar flow past NACA 0012 airfoil at $Re_\infty = 5000$ based on the chord length on the hybrid grid shown in Fig. 22a. Zoomed view of the grid close to the leading edge depicting the viscous padding can be seen in Fig. 22b. This grid consists of 11,669 points, 12,200 cells and 24,275 edges. For this test case, freestream Mach number is $M_\infty = 0.5$ and angle of incidence is 3°. Mach number contours corresponding to LSFD-U and HIFUN-2D solvers are shown is Fig. 22c and d respectively. Computed pressure and friction coefficient distributions are compared with the profiles presented in other numerical work [40] in Fig. 22e. The computed aerodynamic coefficients are compared in Table 6. This test case involves flow separation on the upper surface of the airfoil and the separation point also agrees well with the value reported in Ref. [40]. The presented numerical results amply demonstrate that the LSFD-U solutions are as accurate as the finite volume results and compare well with reference data [40]. From the residual history presented in Fig. 22f, it is clear that both the procedures have similar convergence. The CPU time required for one time step by the LSFD-U and finite volume solvers shown in Table 7, clearly demonstrates the fact that the computational effort associated with the proposed meshless solver is comparable to that of an industry standard finite volume code.

[3] The very purpose of this work is to go beyond such limitations by making use of the LSFD-U solver.

6. Conclusions

This paper presents a methodology to solve the RANS equations within the framework of meshless solvers. It also projects meshless solvers as a viable cartesian grid methodology; in the process, completely circumventing the problem of small cut cells encountered in the finite volume computations. The novelty of the presented methodology lies in the fact that the local structure exhibited by a point distribution obtained from a hybrid cartesian grid is exploited in building a robust and accurate discretization procedure, while retaining the generality a meshless solver enjoys. In fact, this key feature distinguishes the present methodology from conventional least squares based discretization procedures which suffer from the problem of ill-conditioning when employed for highly stretched grids required for the resolution of turbulent boundary layers. The resulting procedure apart from being economical and accurate, is also robust, owing to the positivity of the viscous discretization procedure employed in the critical region of structured grid block employed for resolving the turbulent boundary layer. This feature is expected to be significant in the development of industry standard meshless flow solvers. Also, to the best of the authors knowledge, this work happens to be the first of its kind in presenting the skin friction data from a RANS based turbulent flow computation with in the framework of meshless solvers. Presently the authors are involved in extending these basic ideas to 3D.

Acknowledgments

The authors thank Vikram Sarabhai Space Center (VSSC), Thiruvananthapuram, India for the financial support it provided under the RESPOND program through the schemes Project ISRO 069 for executing the work presented in this paper.

Appendix A. Positivity analysis

In this appendix, we discuss two viscous discretization methodologies as applied to Laplace equation and then test them for their positivity [1,19]. For this purpose, consider the Laplace equation in its divergence form:

$$\frac{\partial f}{\partial x} + \frac{\partial g}{\partial y} = 0, \tag{A.1}$$

where $f = [\phi_x]$, $g = [\phi_y]$ and ϕ is the solution variable. Also, referring to Fig. 2a, note that the unit vector along the ray ij are represented by (n_{x_j}, n_{y_j}).

A.1. Method I

This method draws its inspiration from the conventional LSFD-U procedure [9], wherein, the directional flux $F = fn_{x_j} + gn_{y_j}$, along the ray ij defined at the fictitious interface J (usually placed at the mid-point of the ray ij) is used for determining the flux derivatives at i. In the present case, F_J can be simply defined as,

$$F_J = \frac{\phi_j - \phi_i}{2|\Delta r_{ij}|}. \tag{A.2}$$

Defining a directional flux difference,

$$\Delta F_J = F_J - F_i, \tag{A.3}$$

an estimate of which is given as:

$$\Delta F_J^e = \frac{1}{|\Delta r_{ij}|}\left[\phi_{xx,i}\Delta x_J^2 + 2\phi_{xy,i}\Delta x_J \Delta y_J + \phi_{yy,i}\Delta y_J^2\right]. \tag{A.4}$$

The above equation represents an over determined system and is solved using the method of least squares. The second order derivatives thus obtained from the least squares procedure are used in writing the discrete Laplacian. One distinction of the viscous discretization methodology from its inviscid counterpart is that, F_i involves the definition of solution gradients at node i (not just the solution values). In order to satisfy the consistency of the viscous discretization procedure, these gradients and F_J are required to be determined to second order accuracy [9]. While the use of Eq. (A.2) in computing F_J automatically satisfies this constraint, F_i is computed to second order using a quadratic least squares procedure. Therefore, in this method, the second order derivatives which are already available from the quadratic least squares procedure are ignored, while writing the discrete Laplacian.

A.2. Method II

This method stems from the recognition that a quadratic least squares procedure is already made use of in method I. Therefore, this method simply involves the use of both the first and second order derivatives (second order derivatives alone in the case of Laplacian) obtained from the quadratic least squares procedure in the discretization of the viscous flux derivatives. For very obvious reasons, method II is considerably cheaper compared to method I. In spite of this observation, method I is also considered for analysis, because a positive viscous discretization procedure can make a big difference in terms of the robustness of the resulting flow solver.

Inspired by the work of Coirier [1], we analyse the positivity of the discrete Laplacian obtained using both the above methods. While the Laplacian can be considered as appropriate model for representing the viscous terms of the incompressible flow equations, it may not be adequate for the compressible flow equations. Nevertheless, it is an useful tool in eliminating several possibilities, which may lack the required robustness. In fact, some of the earlier possibilities we had considered for viscous flux discretization, which lacked robustness, turned out to be seriously non-positive [19,24].

For the purpose of analysis, we represent the state update formula at node i, derived from the discrete Laplacian as:

$$\phi_i = \sum_{j=1}^{N} a_j \phi_j, \tag{A.5}$$

where j represents a neighbouring node and N denotes the number of neighbours. The discrete maximum principle demands that,

$$min_j(\phi_j) < \phi_i < max_j(\phi_j) \tag{A.6}$$

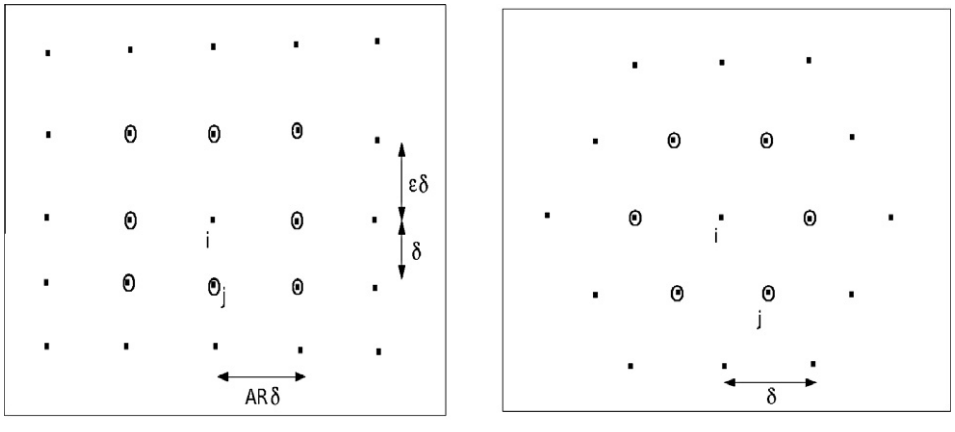
and is satisfied if $a_j \geqslant 0 \forall j$. Following Coirier [1], we introduce a measure of positivity, α_{min} defined as,

$$\alpha_{min} = \frac{min(a_j, 0)}{\sqrt{\sum_{j=1}^{N} \frac{a_j^2}{N}}}. \tag{A.7}$$

It can be seen that $\alpha_{min} = 0$ for a positive scheme.

We choose two model grids for testing the positivity of the viscous discretization schemes. One is a cartesian grid, typically needed for resolving the boundary layers, with stretching along y direction. Definition of the parameters, aspect ratio AR and stretching parameter ϵ are depicted in Fig. 23a. The other is an isotropic point distribution from a regular triangular grid, shown in Fig. 23b. The same connectivity is used for both the methods and the points in the connectivity are shown encircled in Fig. 23. The coefficients a_j's obtained for the cartesian grid with $AR = 100$ and $\epsilon = 1.2$ are presented in Fig. 24 and it can be easily seen that the coefficients satisfy the consistency condition $\sum a_j = 1$. The positivity parameter α_{min} is plotted against the stretching factor ϵ, with AR as a parameter in Fig. 25. From the figure, the superior positivity property of method I is evident. For large values of AR, the positivity parameter also becomes insensitive to changes in AR. Apart from this, the other interesting observation is that α_{min} can show greater sensitivity to ϵ for certain methods. We have made similar observations in the case of finite volume schemes also [24]. This may explain the failure of the CFD codes for values of ϵ beyond, say 1.2 or 1.3 (particularly for turbulent flow computations), while very large values of AR are still acceptable. In any case, for most practical values of AR(say 1000) and ϵ(say 1.2), both the methodologies are non-positive. Both the methods were found positive for the isotropic point distribution [24] and therefore the results are not presented here.

At the implementation level, the effective difference between the methods appear in terms of handling the second derivatives; while an LSFD-U framework is employed in method I (in the sense that the directional derivatives are used), method II simply involves using the second derivatives obtained using the quadratic least squares procedure. A preliminary evaluation of these two methodologies as applied to compressible Navier–Stokes equations did not show any discernible difference either in terms of solution accuracy or in terms of convergence rates [24]. Therefore, considering the cost effectiveness of method II, we have adopted this method for solution update at general points. It is interesting to observe that a better positivity of a discretization scheme (in terms of discrete Laplacian) need not necessarily mean better robustness in



a. Stretched Cartesian point distribution b. Isotropic point distribution

Fig. 23. Model point distribution.

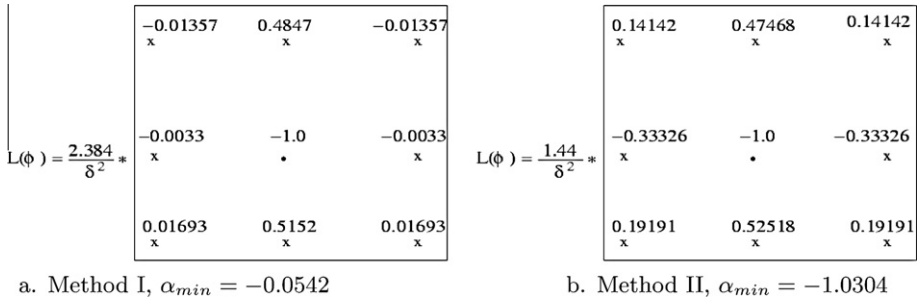


Fig. 24. Discrete Laplacian: stretched cartesian grid ($AR = 100, \epsilon = 1.2$).

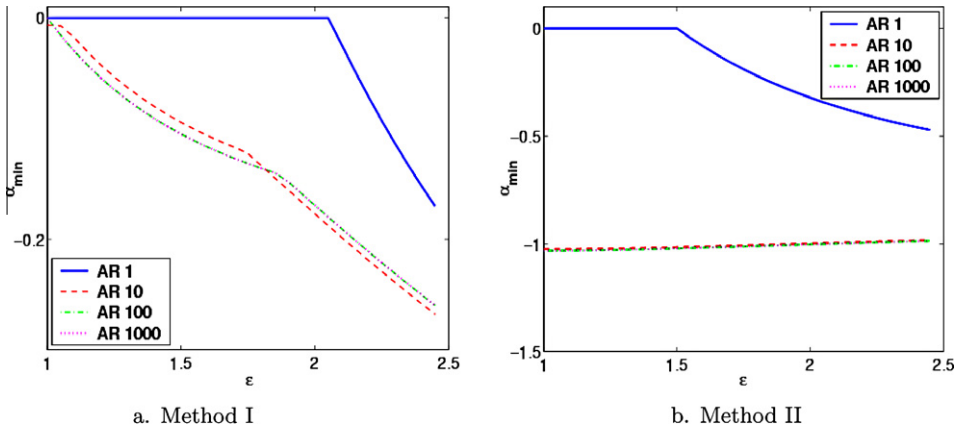


Fig. 25. Positivity parameter α_{min} variation: grid 1.

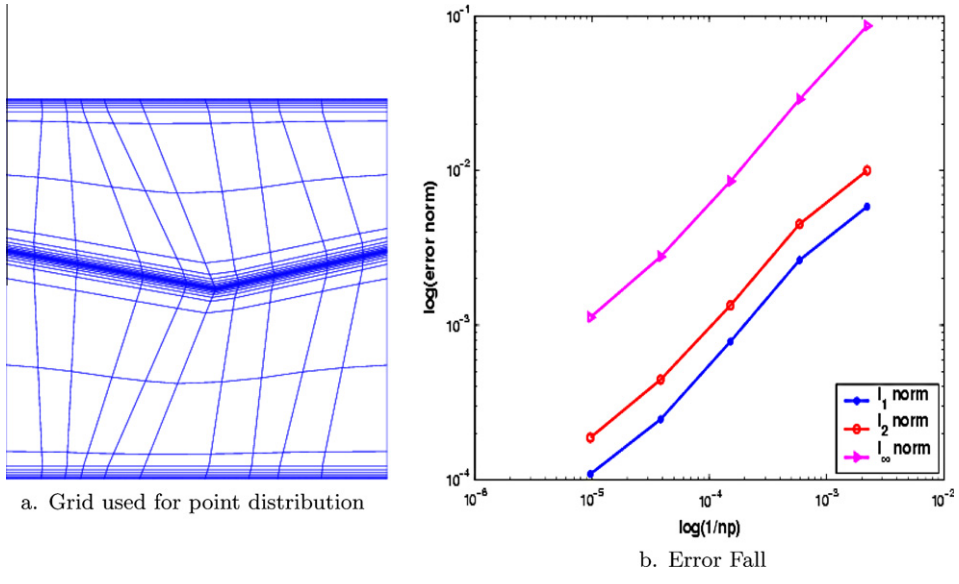
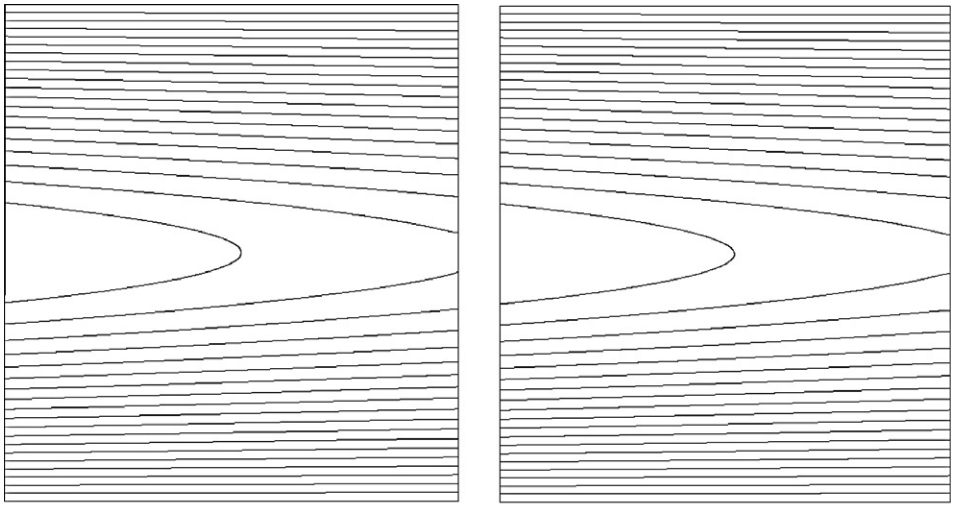


Fig. 26. Global error analysis.

solving for viscous flows. One possible reason for such a behavior could be the inadequacy of the Laplacian in representing the viscous terms of the compressible Navier–Stokes equations. Nevertheless, in our view further studies are required for understanding these contradictions.

Table 8
Error fall rate.

l_1	l_2	l_∞
0.76	0.76	0.81



For Pe = 100: a. Numerical Solution [min=0.0, Δ=0.05,max=1.0]

b. Exact Solution

Fig. 27. Equi ϕ contours.

Appendix B. Error analysis

The viscous discretization procedure proposed for the structured type point (refer to Section 4.1.1) is subjected to global error analysis. The error analysis has been carried out by solving a 2-D scalar convection–diffusion equation [41] given by,

$$\frac{\partial \phi}{\partial t}+\frac{\partial u\phi}{\partial x}+\frac{\partial v\phi}{\partial y}=\alpha\left(\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}\right). \tag{B.1}$$

Here, u, v are convective velocities, α is the diffusion coefficient. This equation can be solved on a rectangular channel with length L and with unit height along with the following boundary conditions:

$$\phi(x,0)=\phi(x,1)=0,$$
$$\phi(0,y)=\sin(\pi y),$$
$$\left.\frac{\partial\phi}{\partial x}\right|_{(L,y)}=0.$$

Considering the cartesian velocity components, $(u,v)=(u_o,0)$, the exact steady state solution is given by:

$$\phi^{exact}(x,y)=\sin(\pi y)\left[\frac{r_2\exp(r_1x+r_2L)-r_1\exp(r_1L+r_2x)}{r_2\exp(r_2L)-r_1\exp(r_1L)}\right], \tag{B.2}$$

where

$$r_{1,2}=\frac{u_o}{2\alpha}\pm\sqrt{\frac{u_o^2}{4\alpha^2}+\pi^2}.$$

An unit square domain ($L=1$) is considered and the equation is solved for a Peclet number $Pe=\frac{Lu_o}{\alpha}$, of 100. Required point distribution is obtained from the grid shown in Fig. 26a. This grid with highly distorted cells and high aspect ratio volumes is representative of the meshes used for adequately resolving the boundary layer. This base grid has 451 points and it is subjected for three levels of refinement. The global error which is defined as follows is monitored with grid refinement:

$$E=\phi^{exact}-\phi^{numerical}. \tag{B.3}$$

Fig. 26b depicts the fall rate of l_1, l_2 and l_∞ norm of the global error with grid refinement. Error norms are plotted against some measure of grid spacing dx, where $dx=\frac{1}{np}$; np being number of points present in the domain. The error fall rate is shown in Table 8. It is encouraging to observe that the scheme is consistent in terms of global error although it is found to be $O(1)$ inconsistent from local analysis. Equi ϕ contours corresponding to the numerical and exact solutions obtained using level 4 grid are shown in Fig. 27. Level 4 grid has 103,201 number of points. These plots clearly demonstrates that the numerical solution converges to the exact solution. This study establishes an empirical evidence to the supraconvergence exhibited by the proposed viscous discretization procedure for structured type points.

References

[1] Coirier WJ. An adaptively-refined, cartesian cell-based schemes for the Euler and Navier–Stokes equations. Ph.D. Dissertation, The University of Michigan; 1994.

[2] Phillippe Guezzaine. An implicit upwind finite volume method for compressible turbulent flows on unstructured meshes. Ph.D. Thesis, Universite De Liege; April 1999.

[3] Delanaye M, Aftosmis MJ, Berger MJ, Liu Y, Pulliam TH. Automatic hybrid-cartesian grid generation for high Reynolds-number flows around complex geometries. AIAA paper, 1999-0777; 1999.

[4] Wang ZJ, Chen RF. Anisotropic solution-adaptive viscous cartesian grid method for turbulent flow simulations. AIAA J 2002;40(10):1969–78.

[5] Partha Mondal, Munikrishna N, Balakrishnan N. Cartesian like grids using a novel grid stitching algorithm for viscous flow computations. J Aircraft 2007;44(5):1598–609.

[6] Batina JT. A gridless Euler/Navier–Stokes solution algorithm for complex Aircraft applications. AIAA paper 1993-0333; 1993.

[7] Ghosh AK, Deshpande SM. Least squares kinetic upwind method for inviscid compressible flows. AIAA paper 1995-1735; 1995.

[8] Balakrishnan N. New least squares based finite difference method. Report no. 99 FM 9, Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India; 1999.

[9] Sridar D, Balakrishnan N. An upwind finite difference scheme for meshless solvers. J Comput Phys 2003;189:1–23.

[10] Morinishi K. An implicit gridless type solver for the Navier–Stokes equations on unstructured meshes. CFD J 2002;9(1).

[11] Morinishi K. A gridless type solver for parallel simulation of compressible flow. In: Lin CA et al., editors. Parallel computational fluid dynamics. Elsevier Science; 1999.

[12] Lohner R, Sacco C, Onate E, Idelsohn S. A finite point method for compressible flow. Int J Numer Methods Eng 2002;53:1765–79.

[13] Kirshman DJ, Liu F. A gridless boundary condition method for the solution of the Euler equations on embedded cartesian meshes with multigrid. J Comput Phys 2004;201:119–47.

[14] Munikrishna N, Balakrishnan N. Turbulent flow computations using meshless solver LSFD-U. In: Seventh Asian computational fluid dynamics conference, Bangalore, India; 2007.

[15] Munikrishna N, Balakrishnan N. Computing turbulent flows using meshless solver LSFD-U. In: Choi et al., editors. Computational fluid dynamics 2008. Berlin, Germany: Springer-Verlag; 2008.
[16] Chung KC. A generalised finite difference method for heat transfer problems of irregular geometries. Numer Heat Transfer 1981;4:345–57.
[17] Harish G, Pavankumar M, Anandhanarayanan K. Store separation dynamics using grid-free Euler solver. AIAA paper 2006-3650; 2006.
[18] Anandhanarayanan K. Applications of grid-free Euler solver to flight vehicles. In: Proceedings of seventh Asian computational fluid dynamics, Bangalore, India; November 26–30, 2007.
[19] Anup Ninawe, Munikrishna N, Balakrishnan N. Viscous flow computations using meshless solver, LSFD-U. In: David Zingg et al., editors. Computational fluid dynamics 2004. Berlin, Germany: Springer; 2004.
[20] Munikrishna N, Karthikeyan N, Balakrishnan N. A meshless solver for computing viscous flows on cartesian like grids. In: Deconinck et al., editors. Computational fluid dynamics 2006. Berlin, Germany: Springer-Verlag; 2006.
[21] Ramesh V, Deshpande SM. Unsteady flow computations for flow past multiple moving boundaries using LSKUM. Comput Fluids 2007;36(10):1592–608.
[22] Athaluri Gnan Kumar. Variants of LSFD-U for compressible flows. M.E. Thesis, Department of Aerospace Engineering, Indian Institute of Science, Bangalore; 2002.
[23] Schlichting H. Boundary layer theory, series in mechanical engineering. Seventh ed. McGraw-Hill; 1979.
[24] Munikrishna N. On viscous flux discretization procedures for finite volume and meshless solvers. Ph.D. Thesis, Department of Aerospace Engineering, Indian Institute of Science, Bangalore; 2007.
[25] Karthikeyan N. Viscous LSFD-U for cartesian grids. M.E. Thesis, Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India; 2005.
[26] Venkatakrishnan V. Convergence to steady state solutions of the Euler equations on unstructured grids with limiters. J Comput Phys 1995;118:120–30.
[27] Deshpande SM. Kinetic meshless method in computational fluid dynamics. In: Proceedings of symposium on advances in fluid dynamics, Jawaharlal Nehru Center for Advanced Scientific Research, Bangalore; July 24–25, 2003.
[28] Chan WM, Steger JL. Enhancements of a three-dimensional hyperbolic grid generation scheme. Appl Math Comput 1992;51(1):181–205.
[29] Munikrishna N, Balakrishnan N. Computing viscous flows on unstructured meshes with hanging nodes. In: Proceedings of 7th annual CFD symposium, CFD division of AeSI, Bangalore, India; August 2004.
[30] Jawahar P, Hemanth Kamath. A high-resolution procedure for Euler and Navier–Stokes computation on unstructured grids. J Comput Phys 2000;164:165–203.
[31] Cockburn Bernardo, Gremaud Pierre-Alain. A priori error estimates for numerical methods for scalar conservation laws. Part II: flux-splitting monotone schemes on irregular cartesian grids. Math Comput 1997;66(218):547–72.
[32] Balakrishnan N, Fernandez G. Wall boundary conditions for inviscid compressible flows on unstructured meshes. Int J Numer Methods Fluids 1998;28:1481–501.
[33] Sridar D, Balakrishnan N. Convergence acceleration of an upwind least squares finite difference based meshless solver. AIAA J 2006;44(10):2189–96.
[34] Roe PL. Approximate Riemann solvers, parameter vectors and difference schemes. J Comput Phys 1981;43:357–72.
[35] Baldwin BS, Lomax H. Thin layer approximation and algebraic model for separated turbulent flows. AIAA paper 1978-257; 1978.
[36] Tritton DJ. Experiments on the flow past a circular cylinder at low Reynolds numbers. J Fluid Mech 1959;6:547–67.
[37] Ratnesh KS, Mahidhar T, Xiaolin Zhong. Very high-order compact finite difference schemes on non-uniform grids for incompressible Navier–Stokes equations. J Comput Phys 2007;224:1064–94.
[38] Cook PH, Mc Donald MA, Firmin MCP. Aerofoil RAE 2822-pressure distribution, boundary layer and wake measurement. Report no. AR 138, AGARD; May 1979.
[39] van den Berg B. Boundary layer measurements on a two dimensional wing with flap. Report no. NLR TR-79009 U, Dutch Aerospace Laboratory (NLR); 1979.
[40] Venkatakrishnan V. Viscous computations using direct solver. Comput Fluids 1990;18(2):191–204.
[41] Olivier-Gooch Carl, Altena Michael Van. A high order accurate unstructured mesh finite volume scheme for the convection–diffusion equation. J Comput Phys 2002;181:729–52.
[42] Ganesh N, Nikhil VS, Balakrishnan N. R-parameter: a local truncation error based adaptive framework for finite volume compressible flow solvers. Comput Fluids 2009;38(9):1799–822.