46th AIAA Aerospace Sciences Meeting and Exhibit
7 - 10 January 2008, Reno, Nevada

AIAA 2008-699

Edge-based Meshless Methods for Compressible Flow Simulations

Aaron Katz [*]

Antony Jameson [†]

Department of Aeronautics and Astronautics, Stanford University, Stanford, CA, 94305, USA

This work concerns the development of a highly accurate and efficient meshless flow solver for inviscid flow in two dimensions. Novel aspects of this work include the application of mesh-based reconstruction, diffusion, and convergence acceleration schemes within an edge-based meshless framework. Most notably, "multicloud," a meshless counterpart to multigrid, has been implemented. Multicloud dramatically enhances convergence to steady state, resulting in a convergence rate which is nearly an order of magnitude faster than single cloud results. Results are presented which indicate good agreement with conventional finite volume results for several test cases, despite the absence of any formal proof of conservation. Correct shock jumps and locations are obtained for airfoils in transonic flow. Lift and drag coefficients also compare well to the finite volume results.

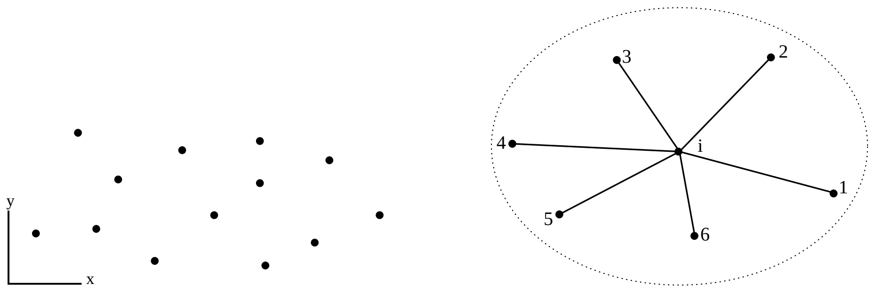
I. Introduction

RECENT progress in the area of meshless methods for CFD computations has shown great promise in terms of accuracy and efficiency.[1–4] Luo et.al.[1] have used a meshless method to compute boundary values within a cartesian mesh framework. Praveen[2] has applied a meshless method to inviscid flow problems using a kinetic/Boltzman discretization. Sridar and Balakrishnan[3] have developed an upwind meshless solver which they demonstrated on subsonic and transonic test cases. An important aspect of their work was an experimental demonstration of mass conservation with the meshless method despite the absence of a formal mathematical proof of conservative properties. Progress in the area of point generation and connectivity for meshless methods has also been made. Löhner and Oñate[4] have proposed a point generation technique to fill a domain an order of magnitude faster than comparable unstructured meshing algorithms. Sridar and Balakrishnan[3] have proposed a connectivity approach which improves the condition number of least squares inversions for meshless schemes.

Despite the progress which has been made, meshless methods have seen little application to practical flow computations. The goal of this paper is to validate a new meshless method against a well-known finite volume code in terms of accuracy and efficiency. The accuracy and efficiency of the present method is obtained primarily via two new aspects to meshless methods, including (1) the application of certain artificial diffusion schemes to meshless methods within an edge-based framework, and (2) the successful implementation of powerful steady state convergence acceleration schemes for meshless methods. Convergence acceleration is mainly attributed to a new "multicloud" algorithm, which parallels multigrid for grid-based CFD. The framework for transfering solutions, residuals, and corrections between point distributions of varying density to efficiently damp high frequency modes is borrowed from conventional multigrid. New definitions for the transfer operators which are suitable for meshless clouds are derived and presented in this work. The combination of accurate diffusion schemes combined with efficient algorithms make the meshless scheme presented here highly competitive compared to mesh-based solvers.

The paper begins by deriving a least squares method for obtaining derivatives on an arbitrary distribution of points. Next, the derivative method is applied to the Euler equations. Discussions regarding steady state integration and multicloud, boundary conditions, and code data structure follow. Finally, results and conclusions of the meshless method are presented.

[*]PhD Candidate, Department of Aeronatics and Astronautics, Stanford, CA, AIAA Member.
[†]Thomas V. Jones Professor of Engineering, Department of Aeronautics and Astronautics, Stanford, CA, AIAA Member.



(a) Arbitrary distribution of points. (b) Least squares cloud

Figure 1. Domain discretization for least squares derivatives

II. Least squares problem for obtaining derivatives

Consider an arbitrary function $\phi(x, y)$ represented by a set of discrete nodal values in a two dimensional domain, as shown in Fig.(1(a)). Furthermore, consider node i surrounded by n nearby nodes as shown in Fig.(1(b)), forming a *cloud* nearest neighbors around i. The function ϕ may be expanded in a truncated Taylor series about i to any of its cloud points as follows:[3]

$$\Delta\phi_{ij} = \sum_{q=1}^{l}\sum_{m=0}^{q}\left(\begin{array}{c} q \\ m \end{array}\right)\frac{\Delta x_{ij}^{q-m}\Delta y_{ij}^{m}}{q!}\frac{\partial^q\phi}{\partial x^{q-m}\partial y^m}, \tag{1}$$

where $\Delta(\cdot)_{ij} = (\cdot)_j - (\cdot)_i$. If $n > \frac{l(l+3)}{2}$, then an over-determined system of equations of the form $A\mathbf{d} = \mathbf{b}$ results, where for a given q,

$$a_q = \left[\frac{\Delta x_{ij}^q}{q!} \quad \frac{\Delta x_{ij}^{q-1}}{q!}\Delta y_{ij} \quad \cdots \quad \frac{\Delta y_{ij}^q}{q!}\right] \tag{2}$$

$$d_q = \left[\frac{\partial^q\phi}{\partial x^q} \quad q\frac{\partial^q\phi}{\partial x^{q-1}\partial y} \quad \frac{q(q-1)}{2}\frac{\partial^q\phi}{\partial x^{q-2}\partial y^2} \quad \cdots \quad \frac{\partial^q\phi}{\partial y^q}\right]^T \tag{3}$$

$$\mathbf{b} = \left[\Delta\phi_1 \quad \Delta\phi_2 \quad \cdots \quad \Delta\phi_n \quad \right]^T \tag{4}$$

The derivatives in $\mathbf{d}$ may be obtained with a least squares procedure invoking the normal equations $A^TA\mathbf{d} = A^T\mathbf{b}$. According to Sridar and Balakrishnan,[3] such a procedure results in an approximation of the *mth* derivative of ϕ to order $h^{l-(m-1)}$, where $m \le l$.

The derivatives in Eq.(1) may always be expressed as weighted sums over the points j in the cloud of i:

$$\frac{\partial^q\phi}{\partial x^{q-m}\partial y^m} \approx \sum_{j=1}^{n} c_{ij}\Delta\phi_{ij}, \tag{5}$$

where the $c_{ij} \sim \frac{1}{\Delta x}$ contain only geometric information and may be obtained from solving the above least squares problem in a preprocess step.

In addition to the above framework, inverse distance weighting may be applied to the least squares problem to better condition the least squares matrix.[5] It has been found that the use of inverse distance weighting results in a system of equations of low enough condition number to solve directly with the normal equations on isotropic meshes. Derivation of a weighted linear least squares formula for first derivatives in two dimensions is given in Appendix B. The least squares procedure of Appendix B follows closely the ideas of least squares gradient reconstruction often used for unstructured meshes. It should be noted that extending the least squares method for derivatives to three dimensions would be easily accomodated by including z terms in the Taylor expansion.

Implicit in the above method for derivatives is a suitable point distribution with cloud definitions of neigboring points for each point in the domain. In the present work, clouds are defined as simply the connectivity of an existing unstructured mesh. By definition, a meshless method should have no need to use any mesh as a means to a solution, but using an unstructured grid for a distrubution of points and connectivity has been a convenient method to focus on the algorithm itself. Future work will address the issue of obtaining point distrubutions and connectivity for meshless methods.

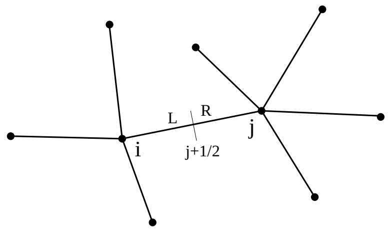


Figure 2. A typical edge in a least squares cloud.

III. Meshless method for the Euler equations

In the last section, a procedure was developed to obtain derivatives of a function from a set of randomly distributed points using a least squares framework. In this section, a least squares discretization for the Euler equations in presented.

Consider the Euler equations in strong conservation law form:

$$\frac{\partial w}{\partial t} + \frac{\partial f}{\partial x} + \frac{\partial g}{\partial y} = 0, \tag{6}$$

where

$$w = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}, \quad f = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ \rho uH \end{pmatrix}, \quad g = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + P \\ \rho vH \end{pmatrix},$$

and

$$E = \frac{P}{(\gamma - 1)\rho} + \frac{1}{2}(u^2 + v^2), \quad H = E + \frac{P}{\rho}.$$

In the above notation, ρ, u, v, P, E, and H are the density, velocity components, pressure, total energy, and total enthalpy.

Like finite difference methods, the meshless algorithm is applied directly to the differential form of the governing equations. Substituting the least squares framework directly into the spatial terms in Eq.(6) at a point i results in

$$\frac{\partial w_i}{\partial t} + \sum_{j=1}^{n} a_{ij} \Delta f_{ij} + \sum_{j=1}^{n} b_{ij} \Delta g_{ij} = 0. \tag{7}$$

It is convenient to define a flux $F = af + bg$ in the direction of the least squares coefficient vector for an edge (a, b), similar to a directional flux through a face area on an unstructured mesh. The approximation of Eq.(7) with the directed flux becomes

$$\frac{\partial w_i}{\partial t} + \sum_{j=1}^{n} \Delta F_{ij} = 0. \tag{8}$$

Eq.(8) represents a non-dissipative, unstable discretization. Stabilization my be conveniently introduced via a properly defined flux function defined at the middle of an edge ij connecting two meshless points, as shown in Fig(2). Using the midpoint flux at $j + \frac{1}{2}$ instead of the flux at j leads to the following discretization of the spatial derivatives,

$$\frac{\partial f}{\partial x} + \frac{\partial g}{\partial y} = 2 \sum_{j=1}^{n} \Delta F_{ij+\frac{1}{2}},$$

where the factor of two is needed since the expansion is only taken to the midpoint of the edge.

A stable algorithm will result if $F_{j+\frac{1}{2}}$ is defined in a manner consistent with the LED criterion of Jameson,[6] such that local maxima cannot increase and local minima cannot decrease. A general framework for stable schemes is obtained by averaging the endpoint fluxes, augmented with diffusive terms:

$$F_{j+\frac{1}{2}} = \frac{1}{2}(F_i + F_j) - \frac{1}{2}d_{j+\frac{1}{2}} \tag{9}$$

Characteristic based diffusion may be computed using the standard Roe matrix[7] with reconstructed left and right states at the edge midpoint, as shown in 2:

$$d_{j+\frac{1}{2}} = \frac{|A(w_L, w_R)|}{2}(w_R - w_L), \tag{10}$$

A discussion of the Roe matrix is given in Appendix A. While the matrix diffusion of Eq.(10) allows for a shock with a single interior point, it is quite expensive to construct compared to scalar schemes. A scheme which is relatively inexpensive with the ability to capture a shock with a single interior point is the CUSP scheme of Jameson.[8] In addition, the CUSP scheme may be formulated to admit constant stagnation enthalpy solutions. The CUSP scheme, which is used in this work, may be expressed as

$$d_{j+\frac{1}{2}} = \alpha^* c(w_R - w_L) + \beta(F_R - F_L). \tag{11}$$

Details of computing the coefficients α^* and β may be found in the work on artificial diffusion schemes by Jameson.[8]

If no reconstruction is performed such that $w_L = w_i$ and $w_R = w_j$, the scheme remains first order accurate. Higher order accuracy may be obtained by reconstructing the solution to the midpoint of each edge in the set of meshless clouds forming the discrete domain. The advantages of an edge-based data structure become apparent in the reconstruction procedure, which parallels closely reconstruction on unstructured meshes. Diffusion may be constructed on an edge-wise basis and distributed to the endpoints of each edge. In this work, higher order accuracy is obtained by using SLIP[6] reconstruction based on least squares gradients as follows:

$$w_L = w_i + \frac{1}{2}\Delta w, \quad w_R = w_j - \frac{1}{2}\Delta w,$$

$$\Delta w = \frac{1}{2}S(\Delta w_i, \Delta w_j)(\Delta w_i + \Delta w_j),$$

$$\Delta w_i = \bar{l} \cdot \nabla w_i, \quad \Delta w_j = \bar{l} \cdot \nabla w_j,$$

where ∇w_i and ∇w_j are computed with the same least squares derivative procedure described in Section 2, and S is a limiter defined as

$$S(u, v) = 1 - \left| \frac{u - v}{|u| + |v|} \right|^q .$$

Thus, if Δw_i and Δw_j are of opposite sign, as in the vicinity of a shock, the limiter S become zero, and the reconstruction reduces to the endpoint states. In the definition of S, the integer q is typically chosen to be 2 or 3. Increasing values of q result in less diffusive schemes.

Finally, the semi-discretized form of the Euler equations may be expressed as a system of ODE's of the form

$$\frac{\partial w_i}{\partial t} + R_i = 0, \tag{12}$$

where

$$R_i = Q_i - D_i, \quad Q_i = \sum_{j=1}^{n} \Delta F_{ij}, \quad D_i = \sum_{j=1}^{n} d_{j+\frac{1}{2}}.$$

Here, the convective (Q) and diffusive (D) portions of the residual(R) are collected separately for reasons explained in the next section.

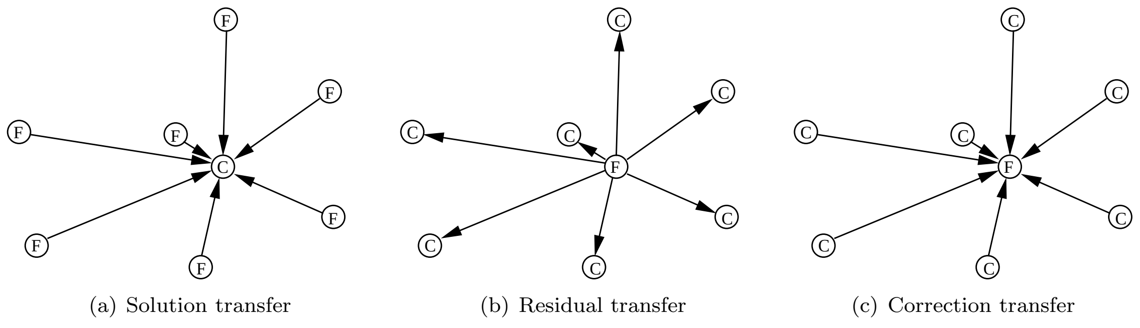


Figure 3. Summary of multigrid transfer operators

IV. Integration to steady state

Equation (12) is a system of ODEs in time, which may be integrated to acheive a steady state. In this work, Jameson's modified Runge Kutta schemes for steady state integration are used.[6] The schemes treat the convective and diffusive portions of the residual separately to acheive a larger region of stability. An m-stage scheme may be expressed as

$$\begin{aligned} w^{(n+1,0)} &= w^n \\ &\vdots \\ w^{(n+1,k)} &= w^n - \alpha_k \Delta t \left(Q^{(k-1)} + D^{(k-1)} \right) \\ &\vdots \\ w^{(n+1)} &= w^{(n+1,m)}, \end{aligned}$$

where

$$\begin{aligned} Q^{(0)} &= Q(w^n), \quad D^{(0)} = D(w^n) \\ &\vdots \\ Q^{(k)} &= Q(w^{(n+1,k)}), \quad D^{(k)} = \beta_k D(w^{(n+1,k)}) + (1-\beta_k) D^{(k-1)} \end{aligned}$$

Local time stepping may be used by employing a local CFL estimate at each node:

$$\Delta t_i = \frac{CFL}{\sum_{j=1}^{n} \left(|a_{ij} u + b_{ij} v| + \sqrt{a_{ij}^2 + b_{ij}^2} c \right)}, \tag{13}$$

where u,v, and c are the local velocities and speed of sound. In practice, $CFL = 5$ was obtained with the above CFL estimate using a five stage scheme.

Along with local time stepping, implicit residual smoothing[9] and enthalpy damping[10] were used to accelerate convergence. Implicit residual smoothing was implemented in a manner similar to that of edge-based unstructured grid algorithms:

$$\bar{R}_i = R_i + \epsilon \nabla^2 \bar{R}_i \tag{14}$$

The solution of Eq.(14) is costly to obtain in general, but two Jacobi iterations were sufficient to increase the CFL by a factor of two.

V. Multicloud

To further accelerate convergence, a "multicloud" framework was developed for the meshless method. Multicloud is the meshless counterpart of multigrid for grid-based CFD. While the details of the solution, residual, and correction transfer operators of the meshless scheme differ from grid based operators, the underlying principle of multigrid holds for multicloud: transfer the problem to a series of successively coarser meshes to damp out unresolvable high frequency modes and communicate the coarse grid corrections back to the fine grid. The multicloud procedure derived here was the most effective tool implemented to enhance the convergence rate of the overall scheme, making the meshless method computationally competitive with grid-based methods.

The alogrithm proceeds by first transfering the flow solution from a fine cloud level, $k-1$, to a coarse cloud level, k, with a solution coarsening operator $T_{k,k-1}$:

$$\mathbf{w}_{\mathbf{k}}^{(\mathbf{0})} = T_{k,k-1}\mathbf{w}_{\mathbf{k-1}}. \tag{15}$$

Likewise, the residuals are transfered to a coarse cloud level with a residual restriction operator $Q_{k,k-1}$. A forcing function, $\mathbf{P}_{\mathbf{k}}$, is computed such that

$$\mathbf{P}_{\mathbf{k}} = Q_{k,k-1}\mathbf{R}_{\mathbf{k-1}}(\mathbf{w}_{\mathbf{k-1}}) - \mathbf{R}_{\mathbf{k}}(\mathbf{w}_{\mathbf{k}}^{(\mathbf{0})}). \tag{16}$$

The forcing function, $\mathbf{P}_{\mathbf{k}}$, represents the difference between the aggregated fine cloud residuals, and the residuals computed with the coarse cloud solution. Subsequently, $\mathbf{R}_{\mathbf{k}}(\mathbf{w}_{\mathbf{k}})$ is replaced by $\mathbf{R}_{\mathbf{k}}(\mathbf{w}_{\mathbf{k}}) + \mathbf{P}_{\mathbf{k}}$ in the time stepping scheme. In this manner, the coarse level iterations are driven by the fine level residuals. At convergence of the fine clouds, the coarse levels do nothing to alter the converged solution. Coarse level iterations proceed by using scalar dissipation to save computational effort, while freezing solid wall and far-field boundary conditions. An iteration on a coarse level results in a corrected solution, w_k^+. Coarse level corrections, based on the difference between the corrected solution and the original solution transfered from the fine grid, are then transfered back to the fine grid with an interpolation-like operator, $I_{k-1,k}$,

$$\mathbf{w}_{\mathbf{k-1}}^{+} = \mathbf{w}_{\mathbf{k-1}} + I_{k-1,k}(\mathbf{w}_{\mathbf{k}}^{+} - \mathbf{w}_{\mathbf{k}}^{(\mathbf{0})}). \tag{17}$$

.

In summary, the multicloud scheme above involves the construction of three operators:

1. a solution coarsening operator, T,
2. a residual restriction operator, Q, and
3. a correction transfer operator, I.

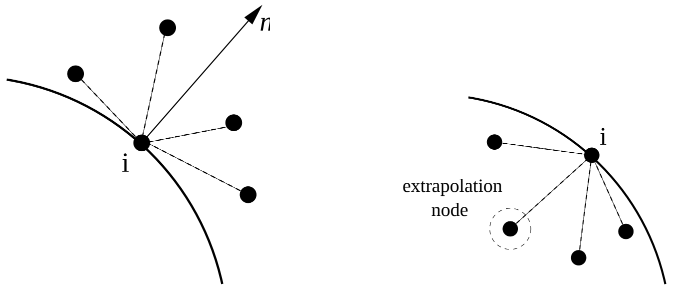
These three operators, shown in Fig.(3) will now be described in detail, completing the description of the multicloud scheme.

All three operators rely on the identification of a nearest neighbor to a point. For the solution coarsening operator, the fine cloud point nearest each coarse cloud point is needed. For the residual restriction and correction transfer operators, the coarse cloud point closest to each fine cloud point is needed. Search operations for nearest neighbors may be performed via a quadtree (octree in three dimensions) approach to avoid an $O(n^2)$ search operation. Once the nearest neighbor has been identified it is a trivial matter to identify its meshless cloud points, which are already determined by other means. The set of points consisting of the nearest neighbor to a point and the meshless cloud of the nearest neighbor will be refered to as the operator cloud.

The solution coarsening operator consists of a weighted sum over the operator cloud for each coarse level point, as shown in Fig.(3(a)). The solution for a coarse level point, i, may be determined by a weighted sum of the operator cloud points j by

$$w_i = \frac{\sum_j c_{ij} w_j}{\sum_j c_{ij}}, \quad c_{ij} = \frac{1}{(\Delta x_{ij}^2 + \Delta y_{ij}^2)^{\frac{q}{2}}}. \tag{18}$$

Thus, T uses normalized inverse distance weights to initialize the solution of each coarse grid point. It is possible to use a taylor expansion from the nearest neighbor to the coarse grid point using least squares gradients of the solution, but the weighted sum of Eq.(18) produces similar results at cheaper cost.



(a) Solid wall boundary (b) Far field boundary

Figure 4. Boundary Conditions.

Unlike the solution coarsening operator, the residual restriction operator is more a scattering than a gathering. The residual at each fine level is scattered to the operator cloud points with particular weights, as shown in Fig.(3(b)). The amount of residual at fine cloud point i scattered to each coarse cloud point j in the operator cloud is

$$\frac{c_{ij} R_i}{\sum_j c_{ij}} \left(\frac{ds_i}{ds_j} \right)^d , \tag{19}$$

where c_{ij} is defined as in Eq.(18), ds is the average edge length for a meshless point, and d is the spatial dimension of the problem. The residuals are scattered with the ds weights similar to node-based residual scattering on a cartesian mesh in which each fine grid node contributes a fraction $\frac{1}{2^d}$ of its residual to the coarse grid problem.

The correction transfer operator is nearly identical to the solution tranfer operator, with the exception that the direction of transfer is coarse to fine instead of fine to coarse, as shown in Fig.(3(c)). A corrected fine cloud solution at node i is computed with a weighted sum over the operator cloud points j by

$$w_i^+ = w_i + \frac{\sum_j c_{ij} (w_j^+ - w_j^{(0)})}{\sum_j c_{ij}}. \tag{20}$$

where c_{ij} is defined as in Eq.(18). Thus all operators are based on simple normalized inverse distance weights, which may be computed in a preprocess step and stored in a linked list for use in transfer subroutines. A convenient aspect of the three operators discussed is that the operator stencils only require the knowledge of a single nearest neighbor. Subsequently, the entire operator cloud is determined by the existing meshless cloud of the nearest neighbor identified.

VI. Boundary conditions

Stable boundary conditions at walls and the far field complete the discretization of Eq.(6). At a solid boundary in inviscid flow, the following four conditions replace the integrated solution, similar to boundary condition enforcement in a finite difference scheme:

$$\frac{\partial P}{\partial n} = -\frac{\rho u_t^2}{R}, \quad \frac{\partial H}{\partial n} = 0, \quad \frac{\partial u_t}{\partial n} = 0, \quad u_n = 0, \tag{21}$$

where P is the pressure at the surface, u_t is the tangential velocity, u_n is the normal velocity, and R is the radius of curvature. The first boundary condition on the normal derivative of the pressure is a direct result of applying the Euler equations in streamline coordinates for steady flow. The second boundary condition on the derivative of the enthalpy holds for the Euler equations in steady flow and a uniform freestream. While the third boundary condition on the tangential velocity is not strictly correct, it completes the description of the state at the boundary to good accuracy. The fourth boundary condition is the no-slip condition.

The first three boundary conditions, which are conditions on derivatives, may be solved via the meshless framework by noting that

$$\phi_i = \frac{\sum_{j=1}^{n} \alpha_{ij}\phi_j - \frac{\partial \phi}{\partial n}}{\sum_{j=1}^{n} \alpha_{ij}}, \tag{22}$$

where ϕ_i is the desired property at the boundary, $\frac{\partial \phi}{\partial n}$ is the derivative given by the boundary condition, and the summation is taken over the cloud for point i. Additionally, $\alpha_{ij} = n_x a_{ij} + n_y b_{ij}$ are the least squares coefficients in the normal direction at i used to compute a normal derivative, as illustrated in Fig.(4(a)).

At the far field, one-dimensional Riemann invariants are used, while enforcing constant stagnation enthalpy and a compressible vortex correction.[11] The vortex correction involves the use of corrected velocities, u_f and v_f, instead of free stream values, u_o and v_o:

$$u_f = u_o + \frac{\Gamma \sin\theta}{R\,(1 - M_o^2(\sin\theta\cos\alpha - \cos\theta\sin\alpha)^2)}$$

$$v_f = v_o - \frac{\Gamma \cos\theta}{R\,(1 - M_o^2(\sin\theta\cos\alpha - \cos\theta\sin\alpha)^2)}$$

Here, Γ is the computed circulation, R and θ are the polar coordinates of the boundary point measured from the quarter-chord, M_o is the freestream Mach number, and α is the angle of attack. Extrapolation from the interior of the domain is accomplished by identifying the cloud point most closely aligned with the normal direction of the boundary point, as illustrated in Fig.(4(b)). Identification of the extrapolation point is performed in a preprocess step for use throughout the computation.

The enforcement of constant stagnation enthalpy at the far field and solid walls produces boundary solutions which admit constant stagnation enthalpy. This is consistent with the H-CUSP[8] scheme used in this work, which also admits constant enthalpy solutions. The result is a scheme which globally preserves constant stagnation enthalpy.

VII. Data structure

The method for obtaining derivatives on arbitrary clouds of points for the meshless scheme is inherently *node-based*. In other words, each node is considered separately in defining a point cloud, least squares coefficients, and derivatives. While this is a convenient framework for deriving the method, there are significant advantages for implementing the scheme in an *edge-based* manner, or in other words, reducing point clouds to a series of edges with each endpoint belonging to the cloud of the opposing endpoint. Point clouds are thus defined implicitly by the edges.

Two reasons for using an edge-based data structure are (1) conservation and (2) ease of coding. First, while formal conservation is not attained by the meshless scheme, edge-basing of the data allows for the similar reciprocity obtained with edge-based finite volume codes. This reciprocity allows for a close approximation of conservation as point clouds approach well-formed configurations like Cartesian or regular triangular meshes. The second reason for using edges–coding efficiency–is largely a practical implementation issue. Quantities may be conveniently computed in loops over edges and accumulated to the end points of each edge. Data structures are simplified and unified since edges always have two endpoints in any number of spatial dimensions, while clouds can have an arbitrary number of points, which differ from cloud to cloud in general. Below is an example of a FORTRAN 90 derived data type for edges:

```
MODULE VARS
IMPLICIT NONE
INTEGER,PARAMETER :: DOUBLE=SELECTED_REAL_KIND(13,200)
TYPE EDGE_TYPE
   INTEGER :: I,J
   REAL(KIND=DOUBLE) :: AIJ,BIJ,AJI,BJI
ENDTYPE EDGE_TYPE
TYPE(EDGE_TYPE),ALLOCATABLE,DIMENSION(:) :: EDGE
END MODULE
```

Note that the data stored for an edge consists of the global indicies of the endpoints and four least squares coefficients–two for each endpoint. The fact that $a_{ij} \neq a_{ji}$ and $b_{ij} \neq b_{ji}$ is a consequence of the fact that the least squares procedure is fundamentally node-based, with each least squares problem uncoupled from the least squares problem of all other nodes in the domain. This results in a lack of formal conservation. The compact data structure above has proven to be an extemely useful and efficient framework in which to implement the meshless method.

VIII. Results and conclusions

In this section a series of experimental test case results are given for airfoils in two-dimensional inviscid flow. For each test case, lift and drag coefficients are compared with FLO 76 unstructured finite volume results on similar point distributions. In order to assess the numerical stability of the least squares procedure, condition numbers for each cloud were computed and found to be $O(1)$ in all cases. Distance weighting was used with a value of $p = 1$ in Eq.(25). In order to eliminate non-physical discontinuous expansions at a sonic line, rounding of eigenvalues approaching zero was performed according to[6]

$$\bar{\lambda} = \frac{1}{2}\left(\epsilon + \frac{\lambda^2}{\epsilon}\right), \quad if \quad \lambda < \epsilon,$$

where $\epsilon = \frac{c}{8}$, and c is the local speed of sound.

To study the convergence rate and the accuracy of drag prediction, a series of mesh refinments was made for two subsonic test cases shown in Tables 1 and 2. The convergence of drag for these test cases shows roughly second order accuracy and reaches zero with reasonable mesh refinement.

The effect of multicloud on convergence is displayed in Fig.(5). An order of magnitude speed up in the number of iterations to convergence is observed by using four cloud levels. While the convergence of residuals is greatly enhanced by the use of multicloud, convergence acceleration of the global quantities is even more dramatic.

The six test cases presented here show clean shock capturing ability with no overshoots. Accuracy in the lift and drag coefficients was obtained to within 3% of FLO 76 results in all cases. In the case of the Korn airfoil, the upper surface is virtually shock free, indicating a high level of accuracy. In the transonic cases, the shock location and magnitude coincide well with the finite volume results.

In conclusion, the meshless method here produces results comparable to the most accurate and efficient unstructured grid solvers for inviscid flow in two-dimensions. The accuracy is obtained with high order reconstruction of diffusion terms in an edge-based framework. The efficiency of the scheme is obtained with convergence acceleration schemes borrowed from mesh-based schemes, such as local time stepping, implicit residual smoothing, enthalpy damping, and multigrid. The results presented here encourage increased use of the meshless scheme for practical flow computation. While it is not clear that meshless methods would be preferred over grid-based methods in all cases, certain scenarios may be well-suited for such methods. These may include portions of a domain representing complex geometry in which a well-defined mesh is difficult to achieve. Future work will focus on applications in which meshless methods appear to be advantageous.

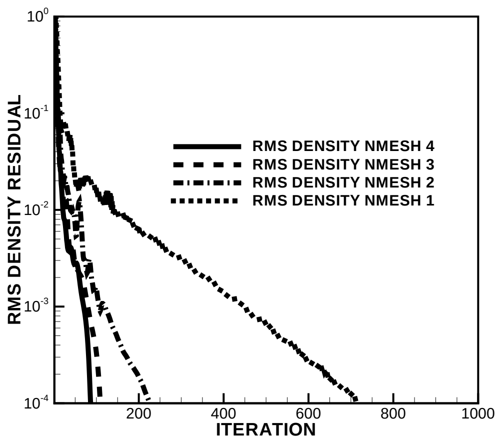
Finally, this work was performed under the support of a National Defense Science and Engineering Graduate (NDSEG) Fellowship. I have benefited greatly from the long term and continuing support of the High Performance Computing Modernization Office of the Department of Defense, the sponsoring agency of my NDSEG fellowship.

Table 1. Convergence of drag for subsonic flow over NACA 0012, $M = 0.50$, $\alpha = 0.0^o$

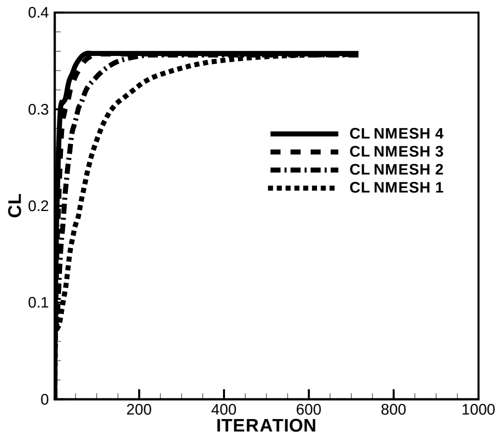
Number of surface points	c_d
20	0.0219
40	0.0022
80	0.0000
160	0.0000

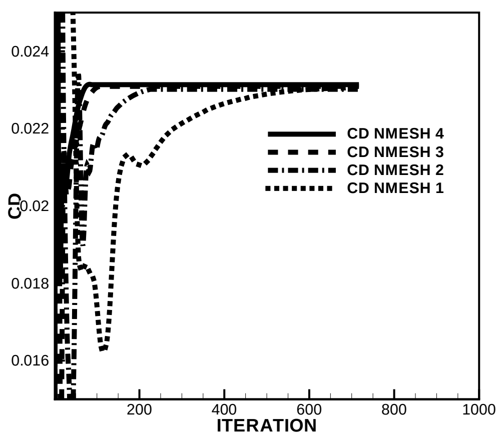
Table 2. Convergence of drag for subsonic flow over NACA 0012, $M = 0.50$, $\alpha = 3.0^o$

Number of surface points	c_d
20	0.0110
40	0.0029
80	0.0015
160	0.0004



(a) RMS density residual





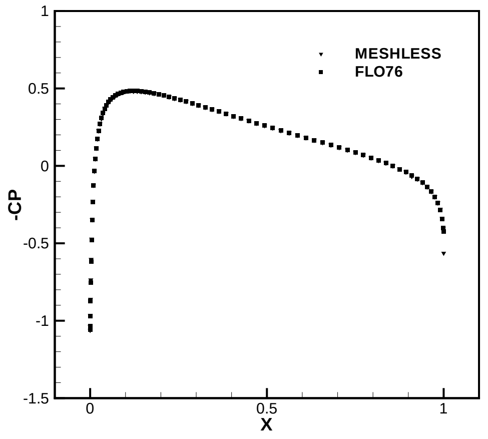
(b) Convergence of lift

(c) Convergence of drag

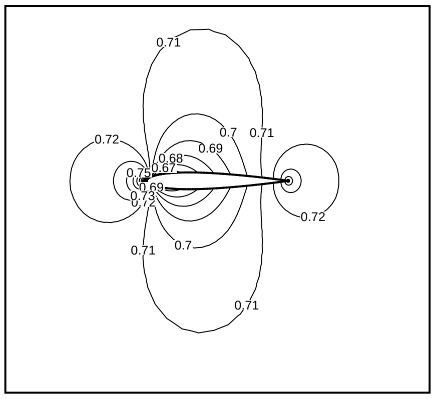
Figure 5. Affect of multigrid on convergence for NACA 0012, $M = 0.80$, $\alpha = 1.25^o$

Table 3. Convergence acceleration summary

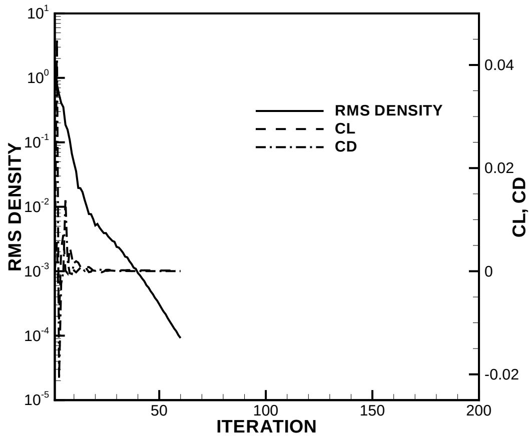
Meshes	Multigrid Cycles	Wall Clock Time (s)
4	87	3.92
3	110	4.58
2	226	7.98
1	717	17.89



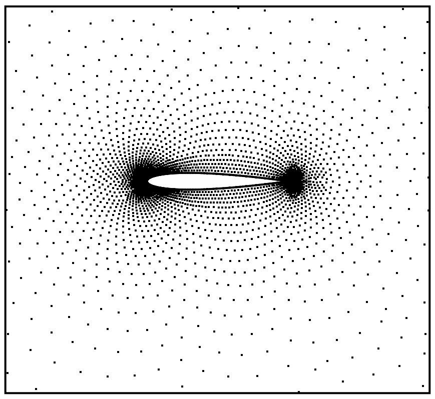
(a) Surface pressure coefficient



(b) Pressure contours



(c) Convergence history

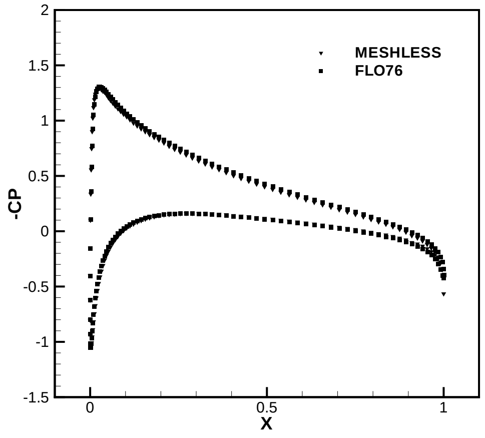


(d) Point distribution

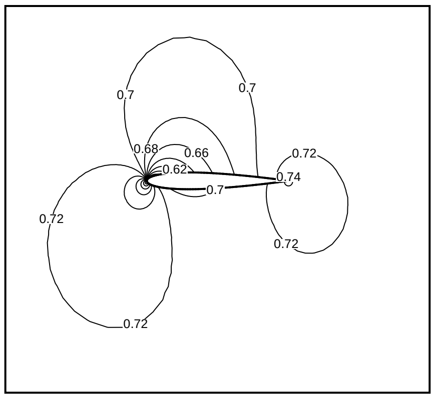
Figure 6. Flow over NACA 0012, $M = 0.50$, $\alpha = 0.0^o$

Table 4. Lift and drag coefficients

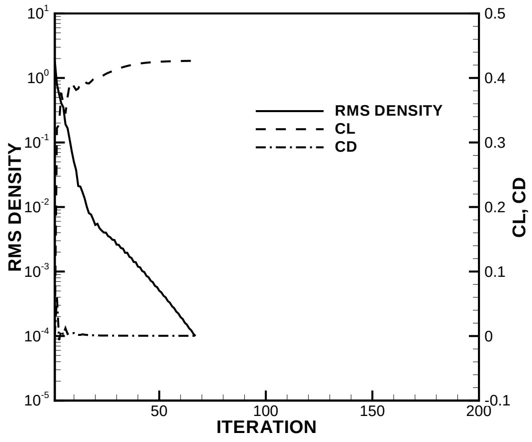
	c_l	c_d
FLO 76	0.0000	0.0002
Meshless	0.0001	0.0000



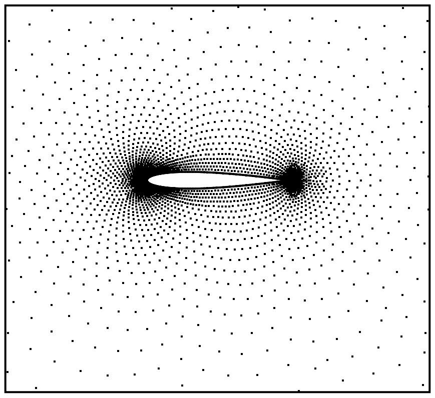
(a) Surface pressure coefficient



(b) Pressure contours



(c) Convergence history

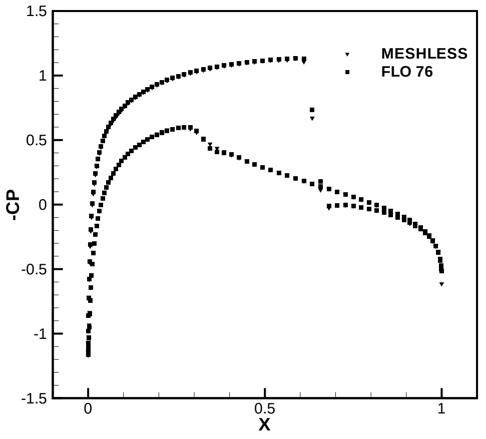


(d) Point distribution

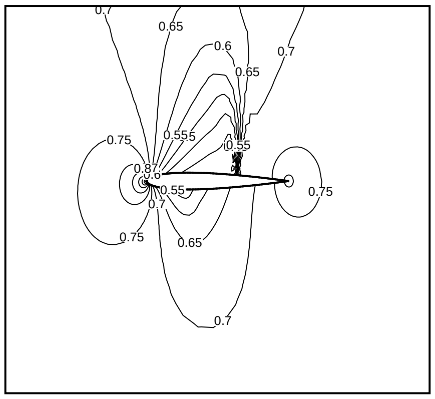
Figure 7. Flow over NACA 0012, $M = 0.50$, $\alpha = 3.0^o$

Table 5. Lift and drag coefficients

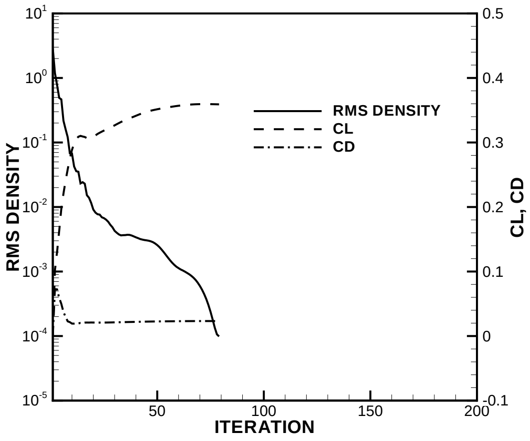
	c_l	c_d
FLO 76	0.4310	0.0002
Meshless	0.4269	0.0004



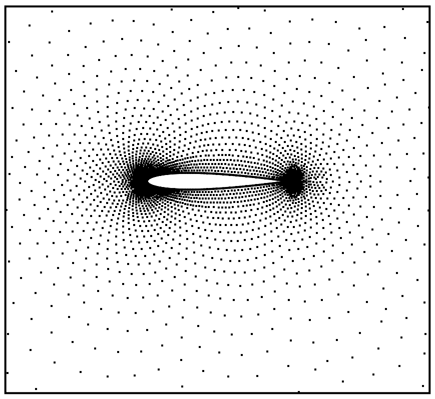
(a) Surface pressure coefficient



(b) Pressure contours



(c) Convergence history

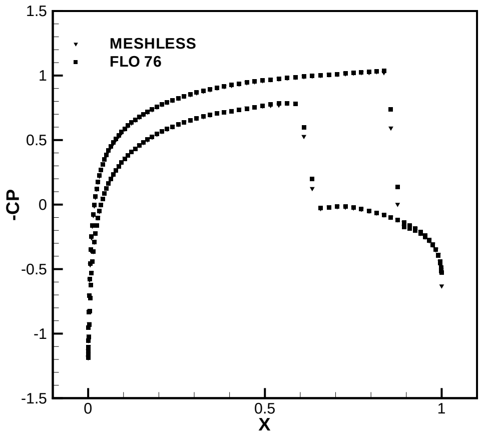


(d) Point distribution

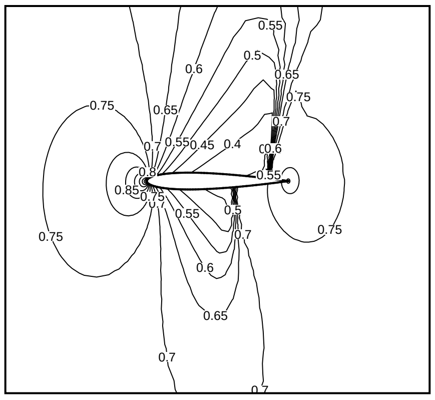
Figure 8. Flow over NACA 0012, $M = 0.80$, $\alpha = 1.25^o$

Table 6. Lift and drag coefficients

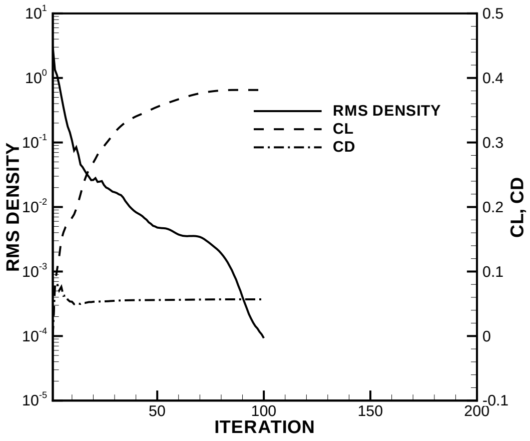
	c_l	c_d
FLO 76	0.3688	0.0238
Meshless	0.3580	0.0231



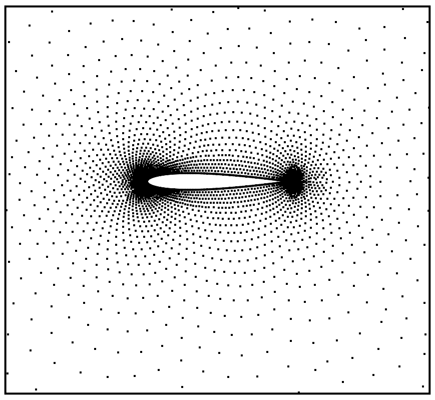
(a) Surface pressure coefficient



(b) Pressure contours



(c) Convergence history

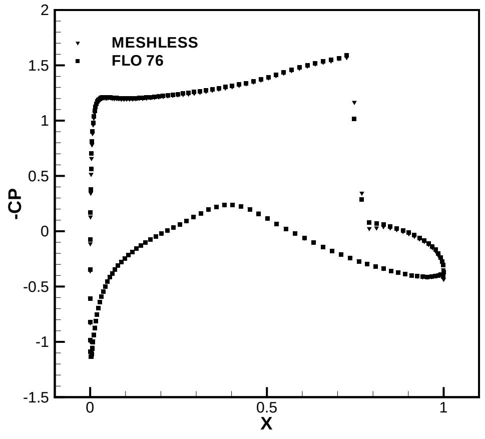


(d) Point distribution

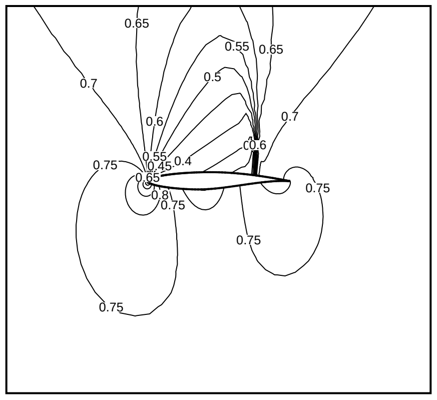
Figure 9. Flow over NACA 0012, $M = 0.85$, $\alpha = 1.0^o$

Table 7. Lift and drag coefficients

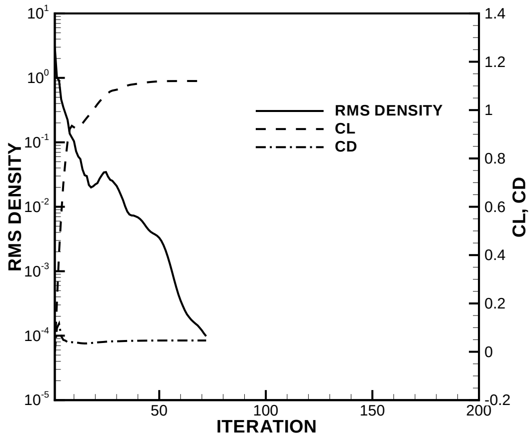
	c_l	c_d
FLO 76	0.3853	0.0584
Meshless	0.3824	0.0570



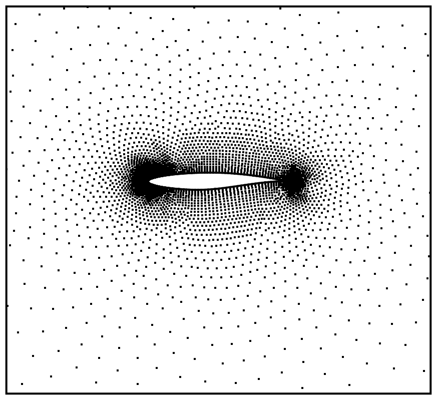
(a) Surface pressure coefficient



(b) Pressure contours



(c) Convergence history

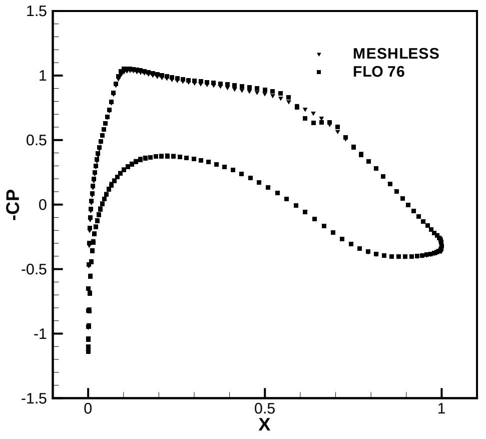


(d) Point distribution

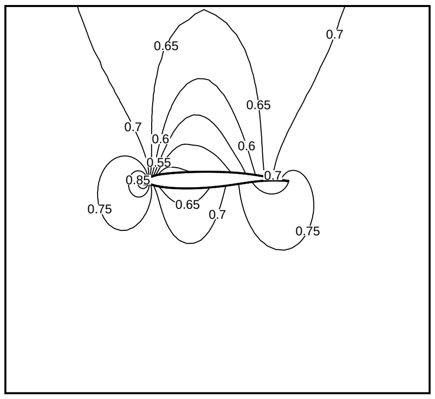
Figure 10. Flow over RAE 2822, $M = 0.75$, $\alpha = 3.0^o$

Table 8. Lift and drag coefficients

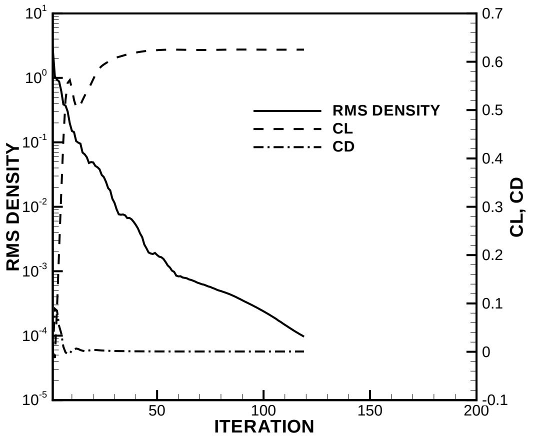
	c_l	c_d
FLO 76	1.1293	0.0470
Meshless	1.1204	0.0468



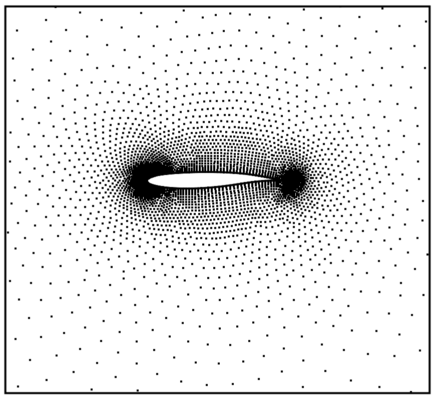
(a) Surface pressure coefficient



(b) Pressure contours



(c) Convergence history



(d) Point distribution

Figure 11. Flow over Korn airfoil, $M = 0.75$, $\alpha = 0.0^o$

Table 9. Lift and drag coefficients

	c_l	c_d
FLO 76	0.6355	0.0000
Meshless	0.6255	0.0005

A. Eigenvector decomposition

In constructing the Roe flux of Eq(10), it is necessary to define $|A| = T|\Lambda|T^{-1}$, where $|\Lambda|$ is a diagonal matrix containing the absolute values of the eigenvalues of A, and the columns of T contain the eigenvectors of A. The mean value Jacobian, $A(w_L, w_R)$, is simply the standard Jacobian evaluated using Roe-averaged variables:

$$u = \frac{\sqrt{\rho_R}u_R + \sqrt{\rho_L}u_L}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \quad v = \frac{\sqrt{\rho_R}v_R + \sqrt{\rho_L}v_L}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \quad H = \frac{\sqrt{\rho_R}H_R + \sqrt{\rho_L}H_L}{\sqrt{\rho_L} + \sqrt{\rho_R}}$$

The two dimensional Jacobian matrix of $F = n_x f + n_y g$ is

$$A = \begin{bmatrix} 0 & n_x & n_y & 0 \\ n_x(\gamma-1)\frac{q^2}{2} - uu_n & u_n - (\gamma-2)n_x u & n_y u - (\gamma-1)n_x v & n_x(\gamma-1) \\ n_y(\gamma-1)\frac{q^2}{2} - vu_n & n_x v - (\gamma-1)n_y u & u_n - (\gamma-2)n_y v & n_y(\gamma-1) \\ u_n\left((\gamma-1)\frac{q^2}{2} - H\right) & n_x H - (\gamma-1)uu_n & n_y H - (\gamma-1)vu_n & \gamma u_n \end{bmatrix}.$$

It follows that A may be diagonalized by $\Lambda = T^{-1}AT$, with

$$\Lambda = \begin{bmatrix} u_n & 0 & 0 & 0 \\ 0 & u_n & 0 & 0 \\ 0 & 0 & u_n + c & 0 \\ 0 & 0 & 0 & u_n - c \end{bmatrix},$$

$$T = \begin{bmatrix} 1 & 0 & 1 & 1 \\ u & cn_y & u + cn_x & u - cn_x \\ v & -cn_x & v + cn_y & v - cn_y \\ \frac{q^2}{2} & c(n_y u - n_x v) & H + cu_n & H - cu_n \end{bmatrix},$$

$$T^{-1} = \begin{bmatrix} 1 - \frac{\gamma-1}{c^2}\frac{q^2}{2} & \frac{\gamma-1}{c^2}u & \frac{\gamma-1}{c^2}v & -\frac{\gamma-1}{c^2} \\ -\frac{un_y - vn_x}{c} & \frac{n_y}{c} & -\frac{n_x}{c} & 0 \\ \frac{1}{2c^2}\left((\gamma-1)\frac{q^2}{2} - cu_n\right) & \frac{1}{2c^2}\left(-(\gamma-1)u + cn_x\right) & \frac{1}{2c^2}\left(-(\gamma-1)v + cn_y\right) & \frac{\gamma-1}{2c^2} \\ \frac{1}{2c^2}\left((\gamma-1)\frac{q^2}{2} + cu_n\right) & \frac{1}{2c^2}\left(-(\gamma-1)u - cn_x\right) & \frac{1}{2c^2}\left(-(\gamma-1)v - cn_y\right) & \frac{\gamma-1}{2c^2} \end{bmatrix},$$

where $u_n = un_x + vn_y$, $q^2 = u^2 + v^2$, and $H = \frac{c^2}{\gamma-1} + \frac{q^2}{2}$.

B. Linear least squares coefficients in two dimensions

The form of Eq(1) for a linear ($l = 1$) fit becomes

$$\Delta\phi_{ij} = \Delta x_{ij}\frac{\partial\phi}{\partial x} + \Delta y_{ij}\frac{\partial\phi}{\partial y}. \tag{23}$$

The weighted least squares problem may be expressed as[2]

$$min\sum_{j=1}^{n} w_{ij}\left[\Delta\phi_{ij} - \Delta x_{ij}\frac{\partial\phi}{\partial x} + \Delta y_{ij}\frac{\partial\phi}{\partial y}\right]^2 \quad wrt \quad \frac{\partial\phi}{\partial x}, \frac{\partial\phi}{\partial y} \tag{24}$$

A simple inverse distance weighting function of the following form may be used:

$$w_{ij} = \frac{1}{(\Delta x_{ij}^2 + \Delta y_{ij}^2)^{p/2}}, \quad p \geq 0. \tag{25}$$

In this work, a value of $p = 1$ was used. Setting up the normal equations in the form of Eqs.(2-4) yields

$$A^T A\mathbf{d} \quad = \quad A^T\mathbf{b} \tag{26}$$

$$\begin{bmatrix} \sum w\Delta x^2 & \sum w\Delta x\Delta y \\ \sum w\Delta x\Delta y & \sum w\Delta y^2 \end{bmatrix} \begin{bmatrix} \frac{\partial\phi}{\partial x} \\ \frac{\partial\phi}{\partial y} \end{bmatrix} = \begin{bmatrix} w_{i1}\Delta x_1 & w_{i2}\Delta x_2 & \cdots & w_{in}\Delta x_n \\ w_{i1}\Delta y_1 & w_{i2}\Delta y_2 & \cdots & w_{in}\Delta y_n \end{bmatrix} \begin{bmatrix} \Delta\phi_1 \\ \Delta\phi_2 \\ \vdots \\ \Delta\phi_n \end{bmatrix} \tag{27}$$

Computing $(A^T A)^{-1}A^T$ leads to the following linear approximation to the derivatives:

$$\frac{\partial\phi}{\partial x} \approx \sum_{j=1}^{n} a_{ij}\Delta\phi_{ij}, \quad a_{ij} = \frac{w_{ij}\Delta x_{ij}\sum w\Delta y^2 - w_{ij}\Delta y_{ij}\sum w\Delta x\Delta y}{\sum w\Delta x^2 \sum w\Delta y^2 - \left(\sum w\Delta x\Delta y\right)^2} \tag{28}$$

$$\frac{\partial\phi}{\partial y} \approx \sum_{j=1}^{n} b_{ij}\Delta\phi_{ij}, \quad b_{ij} = \frac{w_{ij}\Delta y_{ij}\sum w\Delta x^2 - w_{ij}\Delta x_{ij}\sum w\Delta x\Delta y}{\sum w\Delta x^2 \sum w\Delta y^2 - \left(\sum w\Delta x\Delta y\right)^2} \tag{29}$$

References

[1]Luo, H. and Baum, J., "A Hybrid Cartesian Grid and Gridless Method for Compressible Flows," *AIAA paper* 2005-492, AIAA 43rd Aerospace Sciences Meeting, Reno, NV, January 2005.

[2]Praveen, C., *Develpment and Application of Kinetic Meshless Methods for Euler Equations*, Ph.D. thesis, July 2004.

[3]Sridar, M. and Balakrishnan, N., "An Upwind Finite Difference Scheme for Meshless Solvers," *Journal of Computational Physics*, Vol. 189, 2003, pp. 1–29.

[4]Lohner, R. and Onate, E., "An Advancing Front Point Generation Technique," *Commun. Numer. Meth. Engng.*, Vol. 14, 1998, pp. 1097–1108.

[5]Mavriplis, D., "Revisiting the Least-Squares Procedure for Gradient Reconstruction on Unstructured Meshes," *AIAA paper* 2003-3986, AIAA 16th Computational Fluid Dynamics Conference, Orlando, FL, June 2003.

[6]Jameson, A., "Analysis and Design of Numerical Schemes for Gas Dynamics 1 Artificial Diffusion, Upwind Biasing, Limiters and Their Effect on Accuracy and Multigrid Convergence," *International Journal of Computational Fluid Dynamics*, Vol. 4, 1995, pp. 171–218.

[7]Lohner, R., *Applied CFD Techniques: An Introduction Based on Finite Element Methods*, John Wiley and Sons, 2001.

[8]Jameson, A., "Analysis and Design of Numerical Schemes for Gas Dynamics 2 Artificial Diffusion and Discrete Shock Structure," *International Journal of Computational Fluid Dynamics*, Vol. 5, 1995, pp. 1–38.

[9]Jameson, A. and Mavriplis, D., "Finite volume Solution of the Two-dimensional Euler Equations on a Regular Triangular Mesh," *AIAA Journal*, Vol. 24, 1986, pp. 611–618.

[10]Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes," *AIAA paper* 81-1259, AIAA 14th Fluid and Plasma Dynamic Conference, Palo Alto, CA, June 1981.

[11]Pulliam, T., "Efficient Solution Methods for the Navier-Stokes Equations," *Lecture Notes for the Von Karman Institute for Fluid Dynamics Lecture Series*, 1986.