Journal of Computational Physics 228 (2009) 5237–5250

Contents lists available at ScienceDirect

Journal of Computational Physics

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

Multicloud: Multigrid convergence with a meshless operator

Aaron Katz *, Antony Jameson

Department of Aeronautics and Astronautics, Stanford University, Durand Building, 496 Lomita Mall, Stanford, CA 94305, United States

ARTICLE INFO

Article history:
Received 12 November 2008
Received in revised form 5 February 2009
Accepted 9 April 2009
Available online 3 May 2009

Keywords:
Multigrid
Meshless
Meshfree
Gridless
Gridfree
Radial basis functions
Convergence rate

ABSTRACT

The primary objective of this work is to develop and test a new convergence acceleration technique we call multicloud. Multicloud is well-founded in the mathematical basis of multigrid, but relies on a meshless operator on coarse levels. The meshless operator enables extremely simple and automatic coarsening procedures for arbitrary meshes using arbitrary fine level discretization schemes. The performance of multicloud is compared with established multigrid techniques for structured and unstructured meshes for the Euler equations on two-dimensional test cases. Results indicate comparable convergence rates per unit work for multicloud and multigrid. However, because of its mesh and scheme transparency, multicloud may be applied to a wide array of problems with no modification of fine level schemes as is often required with agglomeration techniques. The implication is that multicloud can be implemented in a completely modular fashion, allowing researchers to develop fine level algorithms independent of the convergence accelerator for complex three-dimensional problems.

1. Introduction

The increasing use of computational fluid dynamics for engineering design and analysis demands highly efficient solution methods. To address the need for efficiency, multigrid algorithms have been developed over the past few decades to improve the convergence rate of iterative solvers. Fedorenko [13] formulated first true "multigrid" algorithm for the standard five point discretization of the Poisson equation on a square. He showed that multigrid is truly an $O(N)$ work algorithm. Brandt [5] formulated multigrid algorithms for linear and non-linear PDEs of elliptic and mixed type. His approach was practical, with applicability to aerodynamics problems. Ni [36] first applied multigrid to the Euler equations, including transonic flow applications. Furthering the work of Ni, Jameson [17,18] combined multigrid with a multistage explicit scheme, which was designed to efficiently damp high frequency modes.

While the aforementioned work was applied to structured grids, unstructured multigrid algorithms were also formulated in pioneering works by Perez [40], Löhner and Morgan [31], Mavriplis and Jameson [33], and Lallemand [26]. A major difficulty for unstructured multigrid is efficiently obtaining coarse grids. Several approaches to generating coarse grids have been proposed in the literature. Perez [40] advocated starting with a coarse mesh and refining successively to form fine levels. While this approach is automatic, it often produces low-quality fine meshes, which may not adequately resolve complex geometry. Mavriplis and Jameson [33] used meshes which were independently generated. Unfortunately, this places a burden on the user to generate coarse meshes, requiring search algorithms to define grid transfer operations. The work of Venkatakrishnan and Mavriplis [50] is an example of agglomeration, which automatically fuses fine level volumes to form coarse level volumes. With agglomeration, the fine grid discretization must often be generalized to handle arbitrary polyhedra which appear on the coarse levels. Agglomeration most often requires edge fusing to eliminate multi-faceted coarse volumes. Other fully automated methods of recursive coarsening include the retriangulation algorithm of Chan and Smith

* Corresponding author.
E-mail addresses: ajkatz@stanford.edu (A. Katz), jameson@baboon.stanford.edu (A. Jameson).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.04.023

[6] and the edge collapse algorithm of Crumpton and Giles [8]. In the retriangulation approach, a subset of the fine level nodes are selected for the coarse level. This subset is then triangulated to form a coarse level. In edge collapse, edges are selected on a fine level and "collapsed" to form a node to which the edge neighbors are connected to form larger volumes.

The algorithms above may be categorized as *geometric* multigrid algorithms since they rely on problem geometry in the form of mesh information to obtain coarse levels. Other *algebraic* multigrid approaches have been proposed, which do not explicitly use the problem geometry [43,49,51]. Rather, algebraic multigrid uses the information contained in a system of algebraic equations itself as a basis for reducing the dimension of the problem. Thus no coarse grid definitions or discretizations are needed. Algebraic multigrid may be used as a "black box" solver for problems in which geometric coarsening is difficult or impossible, such as purely discrete problems. Additionally, algebraic multigrid works on linear systems (which may have been derived from non-linear systems), and most often requires the storage of the corresponding large, sparse matrices. There are some indications that the use of geometry when available will produce superior results than a purely algebraic approach [49]. Discussion of the comparative advantages and disadvantages of geometric and algebraic multigrid approaches is, however, beyond the scope of this paper.

The multicloud algorithm described in this work follows a geometric approach, efficiently obtaining coarse levels and operators for arbitrary mesh types. Like agglomeration, edge collapse, and retriangulation, multicloud uses a fully automated coarsening strategy to obtain coarse levels. However, multicloud is unique in that it relies upon a meshless discretization on the coarse levels, which use point clouds in place of a mesh. Thus, we bypass the meshing steps other methods use to generate coarse meshes (e.g. retriangulation, edge fusing, mesh smoothing). While a detailed comparison of the work needed to form the meshless clouds versus the work to produce coarse volumes with other methods is difficult to make, the coarsening strategy we present here is straightforward and simple. Since multicloud is truly meshless, it may be implemented for arbitrary mesh topologies and discretization schemes on the finest level. As with many coarsening strategies, multicloud may be utilized with no modifications to fine level schemes, supporting modularity in code development. Since multicloud is based on a meshless operator, it will likely be unable to reuse fine level discretization routines unless the fine level is also meshless. Consequently, extra development effort to code the meshless discretization for the coarse levels is required in order to implement multicloud. However, this initial effort could potentially save substantial work over time, since in principle multicloud could be used with any type of mesh topology or discretization strategy without further modification.

The paper begins with a formulation of a meshless algorithm based on radial basis functions. Next, the automatic coarsening procedure to obtain point clouds is discussed. Following the coarsening procedure, methods of prolongation and restriction on clouds of points are developed. Finally, two-dimensional results of the Euler equations are presented and major conclusions are drawn. While the results of this work are limited to two dimensions, the power of the algorithm will be fully realized in three dimensions. However, the two-dimensional work represented here has served as a proof of concept for future work.

2. Meshless scheme for coarse levels

The power of multicloud is derived from a robust meshless scheme. While a number of excellent meshless schemes have been proposed in the literature [3,39,37,34,2,38,42,32,48,29,15,30,4,12], their popularity has remained relatively low. This is most likely for two reasons. First, meshless schemes have not been shown to relieve mesh generation difficulties since point cloud generation seems to be nearly as difficult. Consequently, many meshless schemes have used meshes as a starting point for point clouds, which has been a common source of criticism. In this work, point clouds for the meshless scheme are used only on coarse levels for purposes of convergence acceleration. The point clouds are generated automatically and, unlike many previous works, do not represent a mesh in the traditional sense. Second, no meshless scheme has been shown to be discretely conservative. In this work, formal conservation on the coarse levels is irrelevant, since the meshless operator is only used to accelerate convergence. The fine level solution is entirely independent of the coarse level algorithm.

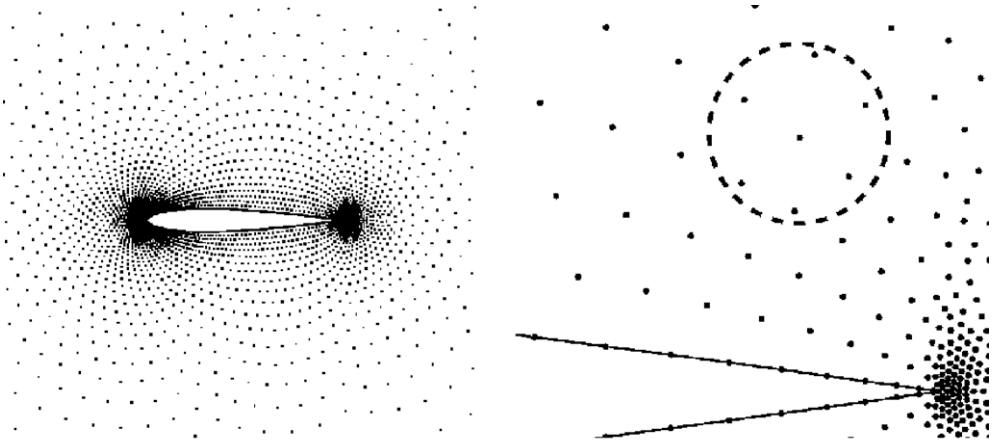
The meshless scheme operating on coarse levels for multicloud is based on radial basis functions applied to the strong form of a given partial differential equation. While radial basis functions approximate the solution on the global domain, shown in Fig. 1(a), they are constructed on local sets of nearest neighbors to a point, called local clouds, shown in Fig. 1(b). In the method described here, partial derivatives on scattered data are reduced to weighted sums over local point clouds. Because the strong form is used, weights depend purely on geometric considerations and may be computed in a preprocess step for use throughout the computation.

A radial basis function is any function which satisfies

$$\theta(\boldsymbol{x}) = \theta(\|\boldsymbol{x}\|), \tag{1}$$

with the norm usually taken as the Euclidean distance. Commonly used radial basis functions include Gaussians, thin plate splines, logarithmics, and multiquadrics. Franke [14] concluded that multiquadrics converged most rapidly and produced the least error of nearly all radial basis functions used for scattered data interpolation at the time. This conclusion has led to widespread use of radial basis functions for the discretization of partial differential equations [23,24,16,41,27,46,44,52,11,10,7,47]. The use of the multiquadric radial basis function is also used here and may be expressed in two dimensions as

$$\theta_i(\boldsymbol{x}) = \sqrt{(x - x_i)^2 + (y - y_i)^2 + \sigma^2}, \tag{2}$$



(a) Global cloud of points. (b) Local cloud of points.

Fig. 1. Global and local cloud representations for the meshless method.

where the basis function is centered at $\boldsymbol{x}_i$, and σ is a free parameter. Unfortunately, there exists no analysis to guide in the selection of σ. In the context of multicloud, σ may be selected to enhance convergence. Setting σ proportional to the average nodal spacing of the local cloud has proven effective.

In the radial basis method, we seek to construct a local approximation, $\hat{\phi}(\boldsymbol{x})$, to an unknown function $\phi(\boldsymbol{x})$. For the Euler equations, the unknown function will be the flux vectors. While $\phi(\boldsymbol{x})$ is unknown, its discrete values are known at the n nodes in the local cloud surrounding node 0. The discrete values are denoted $\phi_i = \phi(\boldsymbol{x}_i)$, where $i = 0, \ldots, n$. The approximating function takes the form

$$\hat{\phi}(\boldsymbol{x}) = \boldsymbol{t}^T(\boldsymbol{x})\boldsymbol{\lambda} + \boldsymbol{p}^T(\boldsymbol{x})\boldsymbol{\alpha}, \tag{3}$$

where $\boldsymbol{t}$ and $\boldsymbol{p}$ are the vectors of radial basis functions and appended polynomials, respectively, and $\boldsymbol{\lambda}$ and $\boldsymbol{\alpha}$ are the corresponding unknown weights for which we seek. The vectors $\boldsymbol{t}$ and $\boldsymbol{p}$ take the form

$$\boldsymbol{t}^T(\boldsymbol{x}) = [\,\theta_0(\boldsymbol{x}) \quad \theta_1(\boldsymbol{x}) \quad \cdots \quad \theta_n(\boldsymbol{x})\,], \quad \boldsymbol{p}^T(\boldsymbol{x}) = [\,p_1(\boldsymbol{x}) \quad p_2(\boldsymbol{x}) \quad \cdots \quad p_m(\boldsymbol{x})\,].$$

The appended polynomials are necessary to ensure a solution to the approximation problem of Eq. (3), as shown by Schaback and Wendlend [45]. For the multiquadric function, monomials up to at least linear terms should be added. In addition, linear terms ensure exact reproducibility of linear functions, as shown by Liu and Gu [28]. This may be seen by expanding the one-dimensional multiquadric function in a Taylor series:

$$\sqrt{c^2 + x^2} = c\sqrt{1 + \tilde{x}^2} = c\left[1 + \frac{\tilde{x}^2}{2} + \frac{\tilde{x}^4}{8} + \frac{\tilde{x}^6}{16} + \cdots\right]. \tag{4}$$

It is clear that the multiquadric basis function omits all odd powers of the independent variable, including the linear terms. Therefore, adding a linear polynomial to the approximation of Eq. (3) enables linear consistency and the reproduction of a linear function exactly. It should also be noted that linear polynomials are needed to accurately represent rigid body translation and rotation for some classes of problems, as in solid mechanics or the mesh deformation algorithm of de Boer et al. [9]. In this work, we set

$$\boldsymbol{p}^T(\boldsymbol{x}) = [\,1 \quad x \quad y\,].$$

The unknown weights, $\boldsymbol{\lambda}$ and $\boldsymbol{\alpha}$, in Eq. (3) may be obtained by invoking the following $n + 1$ equations and m constraints:

$$\hat{\phi}(\boldsymbol{x}_i) = \phi_i, \quad i = 0, \ldots, n, \quad \sum_{i=0}^{n} \lambda_i p_j(\boldsymbol{x}_i) = 0, \quad j = 1, \ldots, m. \tag{5}$$

The resulting system may be recast as

$$\begin{bmatrix} \mathbf{T} & \mathbf{P}^T \\ \mathbf{P} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\alpha} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{\phi} \\ \boldsymbol{0} \end{Bmatrix}, \tag{6}$$

where

$$\mathbf{T}^T = [\,\boldsymbol{t}(\boldsymbol{x}_0) \quad \boldsymbol{t}(\boldsymbol{x}_1) \quad \cdots \quad \boldsymbol{t}(\boldsymbol{x}_n)\,], \quad \mathbf{P} = [\,\boldsymbol{p}(\boldsymbol{x}_1) \quad \boldsymbol{p}(\boldsymbol{x}_2) \quad \cdots \quad \boldsymbol{p}(\boldsymbol{x}_n)\,], \quad \boldsymbol{\phi}^T = [\,\phi_0 \quad \phi_1 \quad \cdots \quad \phi_n\,].$$

Writing the inverse of the matrix in Eq. (6) as

$$\begin{bmatrix} \mathbf{T} & \mathbf{P}^T \\ \mathbf{P} & \mathbf{0} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{A} & \mathbf{B}^T \\ \mathbf{B} & \mathbf{C} \end{bmatrix}, \tag{7}$$

the approximation of Eq. (3) becomes

$$\hat{\phi}(\boldsymbol{x}) = (\boldsymbol{t}^T(\boldsymbol{x})\mathbf{A} + \boldsymbol{p}^T(\boldsymbol{x})\mathbf{B})\boldsymbol{\phi}. \tag{8}$$

The derivatives of $\hat{\phi}$ at $\boldsymbol{x}_0$ may be obtained directly from

$$\frac{\partial \hat{\phi}(\boldsymbol{x}_0)}{\partial x} = \left(\frac{\partial \boldsymbol{t}^T(\boldsymbol{x}_0)}{\partial x}\mathbf{A} + \frac{\partial \boldsymbol{p}^T(\boldsymbol{x}_0)}{\partial x}\mathbf{B}\right)\boldsymbol{\phi} = \boldsymbol{a}^T\boldsymbol{\phi},$$

$$\frac{\partial \hat{\phi}(\boldsymbol{x}_0)}{\partial y} = \left(\frac{\partial \boldsymbol{t}^T(\boldsymbol{x}_0)}{\partial y}\mathbf{A} + \frac{\partial \boldsymbol{p}^T(\boldsymbol{x}_0)}{\partial y}\mathbf{B}\right)\boldsymbol{\phi} = \boldsymbol{b}^T\boldsymbol{\phi},$$

where $\boldsymbol{a}$ and $\boldsymbol{b}$ are vectors of constants independent of ϕ_i. It is only the metric vectors, $\boldsymbol{a}$ and $\boldsymbol{b}$, which are computed and stored in the preprocessing step. Owing to linear reproducibility, the derivatives may be expressed as

$$\frac{\partial \hat{\phi}(\boldsymbol{x}_0)}{\partial x} = \sum_{i=0}^{n} a_i\phi_i = \sum_{i=1}^{n} a_i(\phi_i - \phi_0), \quad \frac{\partial \hat{\phi}(\boldsymbol{x}_0)}{\partial y} = \sum_{i=0}^{n} b_i\phi_i = \sum_{i=1}^{n} b_i(\phi_i - \phi_0). \tag{9}$$

The use of radial basis functions to approximate field variables on scattered data generally requires the inversion of larger matrices than least squares methods [3,39]. The size of the matrix in Eq. (7) is $(n+m+1)$ by $(n+m+1)$, where n is the number of points in the local cloud surrounding node 0, and m is the number of added monomials. However, our numerical experiments indicate that radial basis methods are significantly more stable and less sensitive to local node movement and position than least squares methods. While it is difficult to prove any order of accuracy with radial basis functions, this is not particularly relevant, since the method will only be used on coarse levels within the multicloud scheme.

The partial derivative estimates of Eq. (9) may be substituted directly into the flux derivatives of the Euler equations in strong form,

$$\frac{\partial \boldsymbol{w}}{\partial t} + \frac{\partial \boldsymbol{f}}{\partial x} + \frac{\partial \boldsymbol{g}}{\partial y} = 0. \tag{10}$$

Approximating the spatial derivatives leads to

$$\frac{\partial \boldsymbol{f}}{\partial x} + \frac{\partial \boldsymbol{g}}{\partial y} \approx \sum_{i=1}^{n}(a_i(\boldsymbol{f}_i - \boldsymbol{f}_0) + b_i(\boldsymbol{g}_i - \boldsymbol{g}_0)) = \sum_{i=1}^{n}(\boldsymbol{F}_i - \boldsymbol{F}_0), \tag{11}$$

where $\boldsymbol{F} = a\boldsymbol{f} + b\boldsymbol{g}$ is a directed flux. A stable scheme may be obtained by invoking the Local Extremum Diminishing (LED) principle of Jameson [19]. First order scalar diffusion is sufficient since accuracy is of little concern on coarse levels. A diffusive term, d_i, may be added to each node of a local cloud of the form

$$d_i = |\lambda_{max}|(\boldsymbol{w}_i - \boldsymbol{w}_0), \tag{12}$$

with $\lambda_{max} = |a_i u + b_i v| + c\sqrt{a_i^2 + b_i^2}$, where u, v, and c are the Cartesian velocity components and speed of sound, respectively.

Boundary conditions may be enforced by reflecting interior nodes across normal lines of boundaries and injecting the resulting ghost nodes with solutions consistent with the physics of the problem [3]. For the Euler equations, slip conditions are enforced at solid surfaces, while one-dimensional Riemann invariants are used to solve for the conditions at the far field.

The result of the aforementioned spatial discretization is a coupled system of ordinary differential equations of the form

$$\frac{\partial \boldsymbol{w}_i}{\partial t} + \boldsymbol{R}_i = \boldsymbol{0}, \quad \boldsymbol{R}_i = \boldsymbol{Q}_i - \boldsymbol{D}_i. \tag{13}$$

Here, the convective ($\boldsymbol{Q}$) and diffusive ($\boldsymbol{D}$) portions of the residual ($\boldsymbol{R}$) may be integrated separately using the modified Runge–Kutta scheme of Jameson et al. [21] to effectively damp high frequency error. In addition, local time stepping, enthalpy damping, and implicit residual smoothing may be used to further accelerate convergence.

3. Coarse point cloud distributions

The meshless method of the previous section, which is used on the coarse levels in multicloud, is designed to damp errors from the fine level discretization scheme, which can be a mesh-based scheme. In this work, no assumption is made on the mesh topology of or the specific spatial discretization of the fine level. Instead, a general procedure for obtaining coarse levels of point clouds automatically from any fine level mesh is developed. The coarsening procedure, which is critical to the success of multicloud, follows a two-step process:

(1) determine the subset of fine level points which form the coarse level points, and
(2) form a local cloud of nearby neighbors for each of the coarse level points selected in the previous step.

The procedure may be made mesh transparent by noting that any mesh in any number of spatial dimensions may be expressed as an edge-vertex list. Actually, meshless point clouds may also be expressed implicity using edges if the two vertices forming an edge belong to the local clouds of one another [25]. This fact enables the two-step coarsening procedure described above to be applied recursively.

The first step of the coarsening procedure is illustrated in Fig. 2(a), and is similar to the algorithm of Chan and Smith [6] to color a maximally independent set of vertices. Each fine level point is first initialized as "valid." The list of fine level points is then traversed. If a fine level point is not blanked, the nearest neighbors in its local cloud are blanked. The next point is then tested, until all the points have been traversed and labeled "valid" or "blanked." The set of valid points is not independent of the order in which the points are traversed. In order to preserve the domain boundaries as much as possible, the boundary nodes are traversed first, followed by the interior nodes. For two-dimensional boundaries, this results in the selection of every other fine level node for the coarse level boundary definition. The result of this simple procedure is a subset of the fine level nodes, illustrated as the filled nodes in Fig. 2(a), whose nearest neighbors are all blanked. This subset of nodes forms the next coarse level.

The second step in the coarsening procedure, illustrated in Fig. 2(b), departs from the method of Chan and Smith [6]. Instead of triangulating the coarse level points, we group them into local clouds to apply the radial basis method. While most local cloud definition procedures have focused on nearest neighbor search methods, such as quadtrees and bin searches [48,32,1], these methods can incur significant overhead for all but the simplest problems. A simple and much cheaper approach based on edges is taken here. An edge in the meshless sense is simply a connection between two nodes which contain each other in their local clouds. A global list of edges completely defines local cloud definitions. For each blanked point on the fine level, the valid nearest neighbors in its local cloud are connected to form an edge for the coarse level. It is possible that the blanking procedure performed in the first step could result in a single valid point in the local cloud of a blanked point on the fine level, as shown in Fig. 3. In this case, the status of the blanked point is changed to "valid," and an edge between this

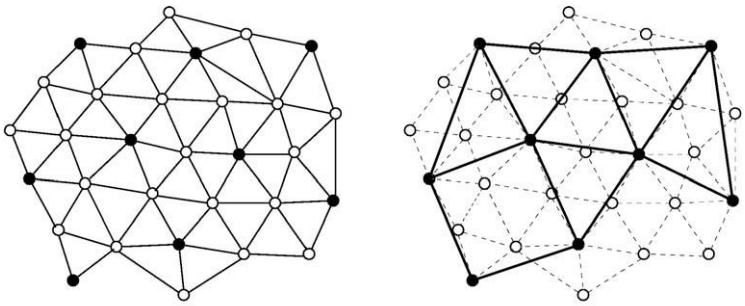


Fig. 2. Point coarsening procedure.

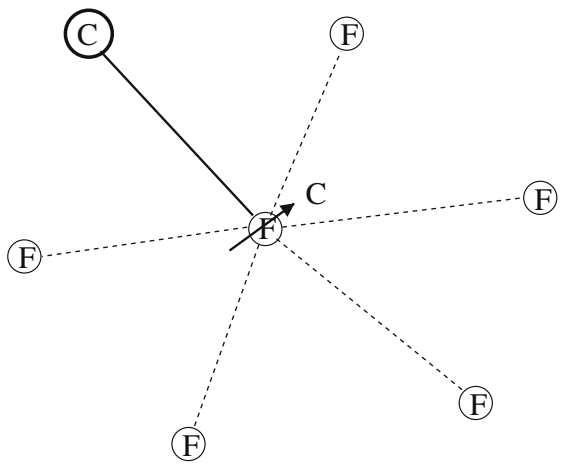


Fig. 3. Changing the status of a fine level point when only one valid point lies in its local cloud.

new valid point and the single valid nearest neighbor is added to the coarse level. The above procedure is much simpler and faster than agglomeration techniques and is entirely mesh transparent. For cases tested so far the procedure has shown to be robust, producing good quality point distributions up to five levels deep.

4. Prolongation and restriction operations

Of course, suitable coarse levels upon which to apply the meshless method are only part of the multicloud framework. Additionally, prolongation and restriction operators between successive levels must be defined. In the context of multicloud, the finest level, which is actually a mesh, is considered a collection of point clouds and treated exactly the same as the coarse levels. This establishes a common method of transfer for all levels. The only difference present on the fine mesh is in the case of a cell-centered scheme, in which nodal values of the solution and residuals should first be obtained via conservative volume weighting procedures.

The multicloud algorithm proceeds along the lines of the Full Approximation Storage (FAS) algorithm of Brandt [5]. First, the solution variables are restricted from a fine cloud level, $k-1$, to a coarse cloud level, k, with a solution coarsening operator, $T_{k,k-1}$:

$$\boldsymbol{w}_k^{(0)} = T_{k,k-1}\boldsymbol{w}_{k-1}. \tag{14}$$

Likewise, the residuals are restricted to a coarse cloud level with a residual coarsening operator $Q_{k,k-1}$. A forcing function, $\boldsymbol{P}_k$, is computed such that

$$\boldsymbol{P}_k = Q_{k,k-1}\boldsymbol{R}_{k-1}(\boldsymbol{w}_{k-1}) - \boldsymbol{R}_k(\boldsymbol{w}_k^{(0)}). \tag{15}$$

The forcing function, $\boldsymbol{P}_k$, represents the difference between the aggregated fine cloud residuals, and the residuals computed with the coarse cloud solution. Subsequently, $\boldsymbol{R}_k(\boldsymbol{w}_k)$ is replaced by $\boldsymbol{R}_k(\boldsymbol{w}_k) + \boldsymbol{P}_k$ at each stage of the iterative time stepping scheme. In this manner, the coarse level iterations are driven by the fine level residuals. At convergence on the fine mesh, the coarse levels do nothing to alter the converged solution.

An iteration on a coarse level results in a corrected solution, $\boldsymbol{w}_k^+$. Coarse level corrections, based on the difference between the corrected solution and the original solution transferred from the fine level, are then interpolated back to the fine grid with a prolongation operator, $I_{k-1,k}$,

$$\boldsymbol{w}_{k-1}^+ = \boldsymbol{w}_{k-1} + I_{k-1,k}(\boldsymbol{w}_k^+ - \boldsymbol{w}_k^{(0)}). \tag{16}$$

The three operators, T, Q, and I, illustrated in Fig. 4, will now be described in detail. First, the solution coarsening operator, T, is simply an injection of the fine level solution to coarse level points since all coarse level points are coincident with fine level points. This is a consequence of the fact that coarse level points are subsets of fine level points. The injection process is shown in Fig. 4(a).

The other two operators involve coincident and non-coincident points. It can be seen from Fig. 2 that all fine level points are either coincident with a coarse level point, or have at least two coarse level points in their local cloud. This is a consequence of our coarsening and cloud definition procedures. The coarsening procedure selects a subset of the fine level points for the coarse level, resulting in many fine level points that coincide with coarse level points. Furthermore, the cloud definition procedure connects two coarse level points which reside in the local cloud of a common fine level point. As explained, if there is only a single coarse level point in the cloud of a given blanked fine point, the fine point is then selected for the coarse level, making it coincident with a coarse level point. Therefore, all fine level points are either coincident with a coarse level point, or have at least two coarse level points in their local cloud.

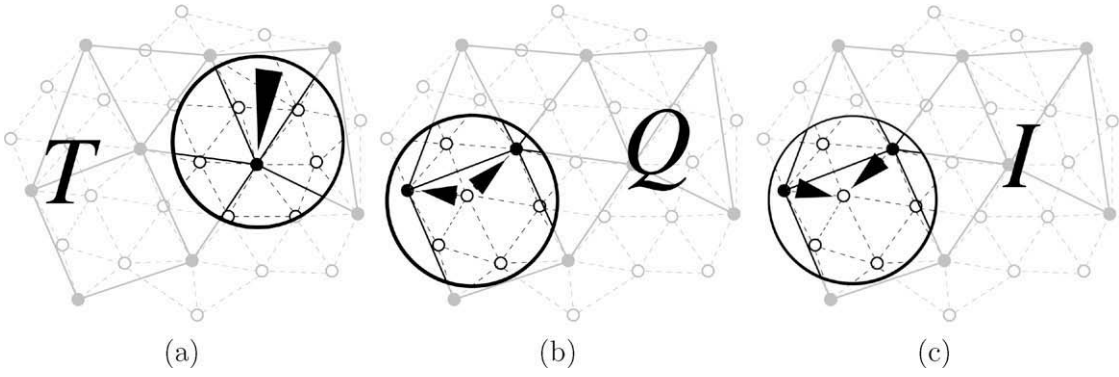


Fig. 4. Illustration of multicloud transfer operators.

For fine level points, i, coincident with or neighboring a coarse level point, j, in its local cloud, inverse distance weight coefficients may be defined:

$$c_i = \begin{cases} |\boldsymbol{r}_i - \boldsymbol{r}_j|^{-1} & \text{for } i \neq j \text{ (not coincident)}, \\ 1 & \text{for } i = j \text{ (coincident)}. \end{cases} \tag{17}$$

Both the residual transfer operator and the correction transfer operator rely on the distance weights of Eq. (17). While we could have used the radial basis function of Eq. (2) for transfer operations, our numerical experiments have shown that the simpler distance method based on Eq. (17) works equally well for this particular case. Therefore, we have chosen to transfer residuals, solution variables, and corrections using convex combinations of surrounding values with weights based on Eq. (17).

The residual transfer operator involves distributing the residual at each fine level point to the valid coarse level points in its local cloud with a weighting, as shown in Fig. 4(b). The aggregated coarse level residual at node j, denoted $(Q\boldsymbol{R})_j$, may be obtained from the coincident and surrounding fine level residuals, $\boldsymbol{R}_j$ and $\boldsymbol{R}_i$, from

$$(Q\boldsymbol{R})_j = \alpha_j \boldsymbol{R}_j + \sum_i \beta_i \boldsymbol{R}_i, \tag{18}$$

where $\alpha_j = (ds_{fj}/ds_{cj})^2$ is the ratio of the average edge lengths of fine and coarse clouds at the coincident node, j, and

$$\beta_i = \left(\frac{1-\alpha_j}{\sum_k c_k}\right) c_i \tag{19}$$

is a normalized inverse distance weight. The residual at the coincident node is scaled with the ratio of the average edge lengths to obtain the correct amount of residual on the coarse level.

We can gain insight into the meshless residual transfer operator by considering nodal restriction for a Cartesian mesh, shown in Fig. 4. Typically for a nodal scheme on a Cartesian mesh, the weight on the coincident node would be 1/4, which would be the value of α_j. Also, from Fig. 4, it can be seen that the corner nodes, which are further away from the side nodes, would carry less weight, contributing 1/16 instead of 1/8 of their residuals. The inverse distance weights of Eq. (17) attempt to mimic this behavior for the Cartesian mesh. Additionally all the weights influencing a coarse level node in the figure sum to one, forming a partition of unity, which forms the basis for many meshless schemes [12]. We desire to maintain the partition of unity for the meshless transfer, which can be verified by noting that $\alpha_j + \sum_i \beta_i = 1$ for the transfer coefficients.

The prolongation operator is similar to the residual restriction operator, with the exception that the direction of transfer is coarse to fine instead of fine to coarse, as shown in Fig. 4(c). A corrected fine cloud solution at fine level node i is computed with a weighted sum over its valid nearest neighbor coarse level points, j, by

$$\boldsymbol{w}_i^+ = \boldsymbol{w}_i + \frac{\sum_j c_j (\boldsymbol{w}_j^+ - \boldsymbol{w}_j^{(0)})}{\sum_k c_k}. \tag{20}$$

If the fine level point, i, is coincident with a coarse level point, j, the corrections are a simple injection since $c = 1$ for coincident points.

It can be seen that all coefficients are based on weights which may be computed in a preprocess step and stored in a linked list for use in transfer subroutines. It should be noted that no search algorithms are used in either the coarsening procedure or any of the multicloud transfer operators. This makes the multicloud procedure extremely fast to set up for an arbitrary mesh (see Fig. 5).

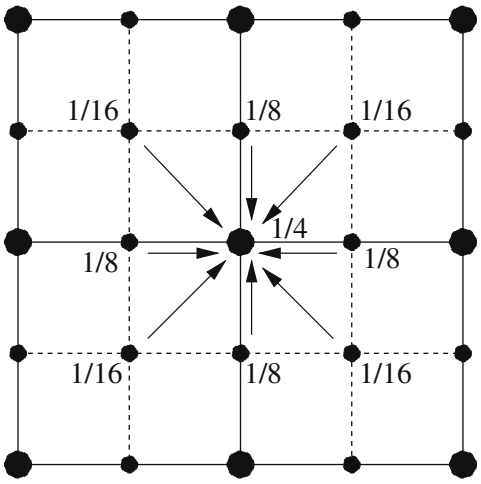


Fig. 5. Node centered residual restriction weights for a Cartesian mesh.

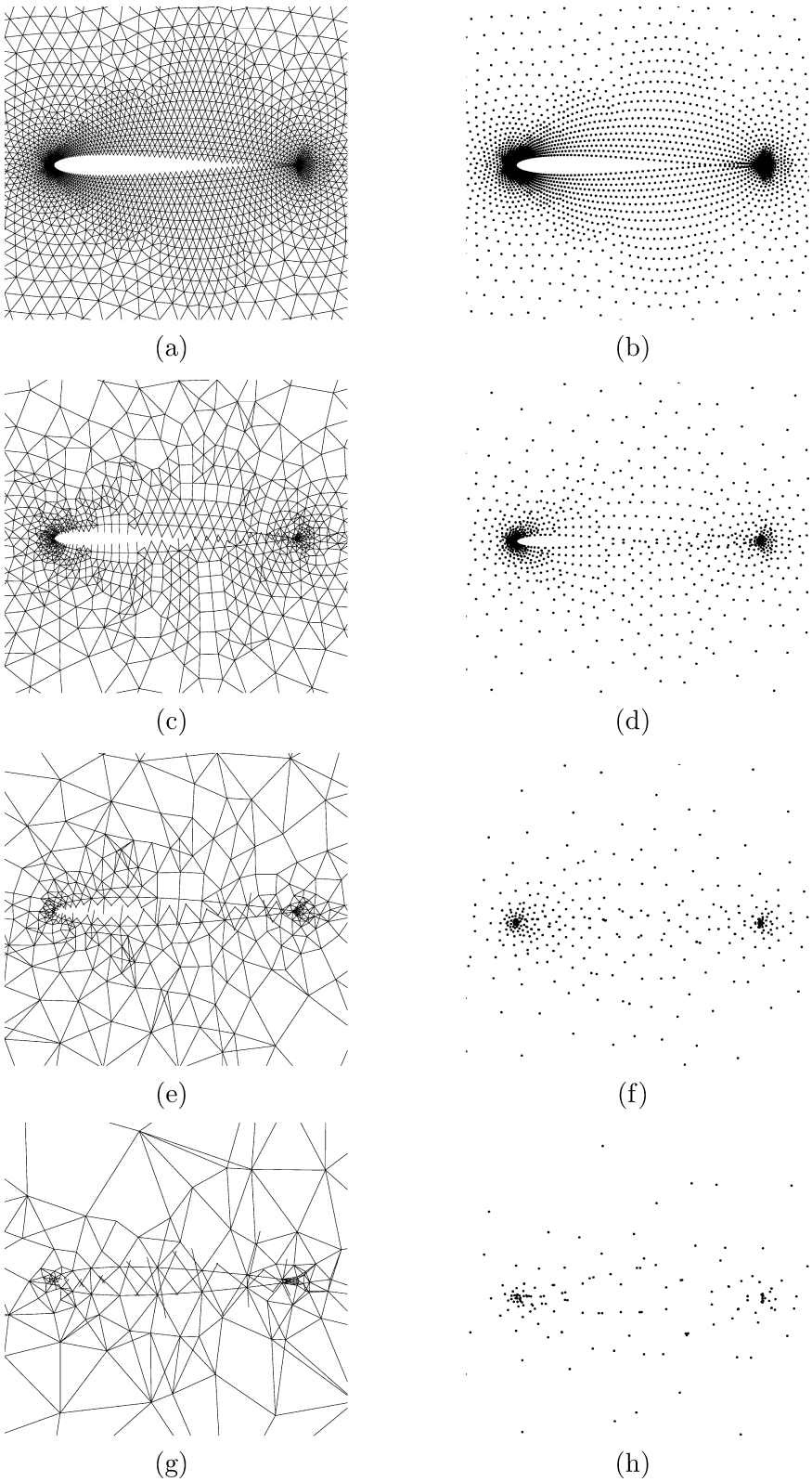


Fig. 6. Results of coarsening procedure for NACA 0012, showing three coarsened cloud levels and corresponding edge connectivity. Number of nodes: level 1 – 5903; level 2 – 1941; level 3 – 728; level 4 – 282.

5. Results

To highlight the capabilities of multicloud, two-dimensional Euler computations for airfoils are given. First, the results of the automatic coarsening procedure for a NACA 0012 airfoil containing 5903 nodes is shown in Fig. 6. The fine level is an unstructured mesh obtained from the *delaundo* package [35]. The coarsened global point distributions are shown in the right column, while the edge connectivity, which defines local clouds is shown in the left column. Levels 2–4 are not meshes, but collections of edges. A close inspection of the coarse level edges shows edges crossing and other features unacceptable for a mesh. This, however, presents no problem for the meshless method, which easily accommodates this more general definition of connectivity. It should be noted that the number of nodes coarsens by roughly a factor of three between successive levels for this case. This is significantly lower than the ideal coarsening factor of four obtained from structured coarsening. While not ideal, the efficient damping of the meshless operator seems to compensate for the non-ideal coarsening, resulting in excellent convergence rates, as will be shown. However, it would be desirable to more closely obtain the ideal coarsening to save on memory. Future research will focus on obtaining more ideal coarsening rates.

Multicloud was used to accelerate three fine level discretization schemes, illustrated in Fig. 7(a)–(c). While the spatial discretization varied among these three schemes, the method of achieving steady state was held constant to test the multicloud convergence properties. In all cases, the modified Runge–Kutta scheme of [21] was used in conjunction with multicloud. The first scheme tested was a high resolution meshless (HRM) scheme based on the method of Katz and Jameson [25],

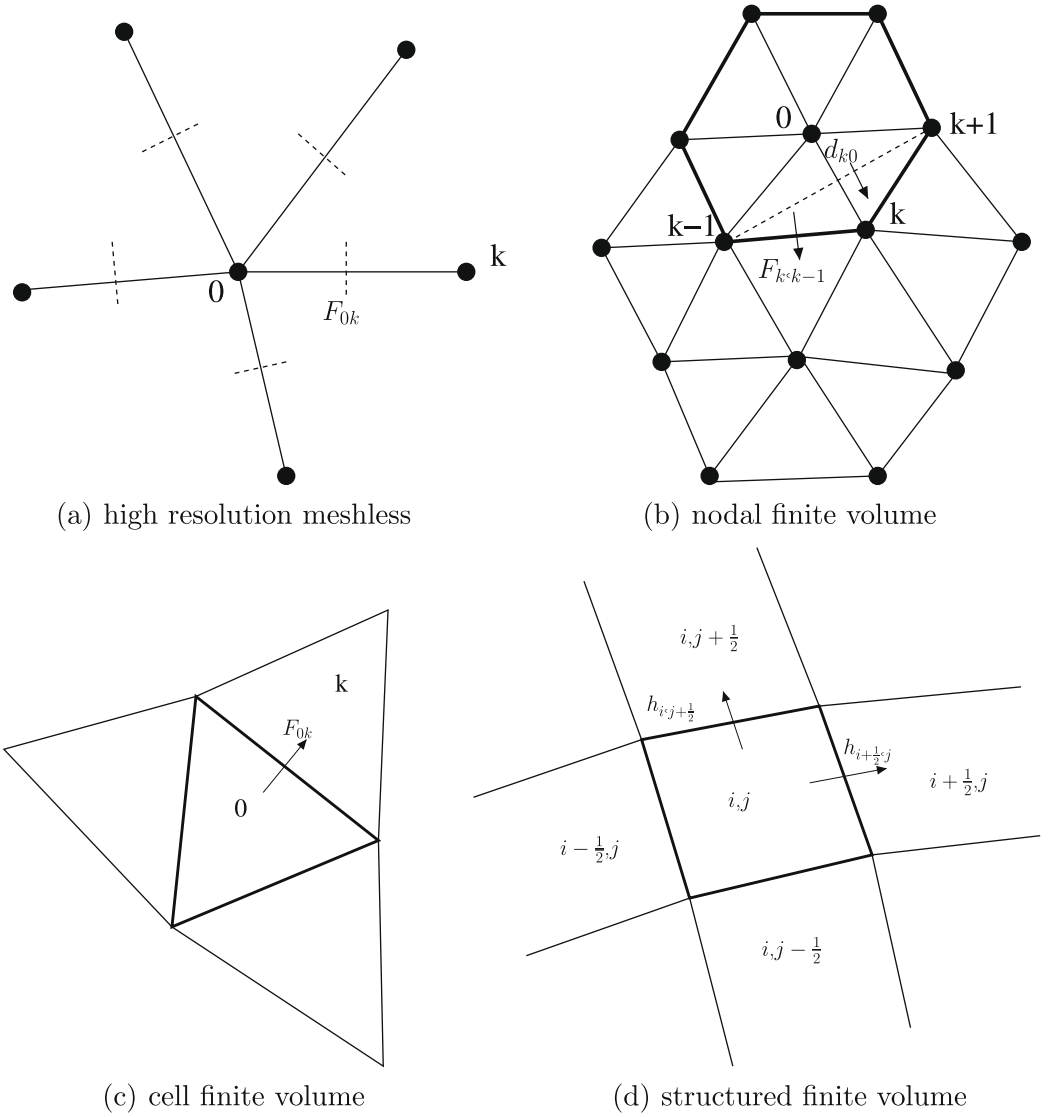


Fig. 7. Illustration of multicloud test schemes.

as shown in Fig. 7(a). The semi-discrete form of the HRM scheme at node 0 with local surrounding nodes, k, may be expressed as

$$\frac{d\boldsymbol{w}_0}{dt} + \sum_k (\boldsymbol{F}_{0k} - \boldsymbol{F}_0) = 0, \tag{21}$$

where $\boldsymbol{F} = a\boldsymbol{f} + b\boldsymbol{g}$ is the flux projected in the direction of the radial basis weights, a and b, of Eq. (9). Here, the edge midpoint flux may be defined as

$$\boldsymbol{F}_{0k} = \frac{1}{2}(\boldsymbol{F}_0 + \boldsymbol{F}_k) - \frac{1}{2}\boldsymbol{d}_{0k}, \tag{22}$$

where $\boldsymbol{d}_{0k}$ is a diffusive flux based on the CUSP scheme of Jameson [19].

The second fine level scheme which used multicloud was a nodal finite volume (NFV) scheme, illustrated in Fig. 7(b). The NFV scheme was implemented similar to the unstructured scheme of Jameson [19], in which nodal volumes are taken to be the union of triangles sharing a common node. For a node 0 surrounded by neighboring nodes k, as defined by the triangulation, the semi-discrete form may be expressed as

$$V_0 \frac{d\boldsymbol{w}_0}{dt} + \sum_k (\boldsymbol{F}_{k,k-1} - \boldsymbol{d}_{k0}) = 0, \tag{23}$$

where the directed flux, $\boldsymbol{F}_{k,k-1}$, is defined as

$$\boldsymbol{F}_{k,k-1} = \frac{1}{2}(\boldsymbol{g}_k + \boldsymbol{g}_{k-1})(x_k - x_{k-1}) - \frac{1}{2}(\boldsymbol{f}_k + \boldsymbol{f}_{k-1})(y_k - y_{k-1}).$$

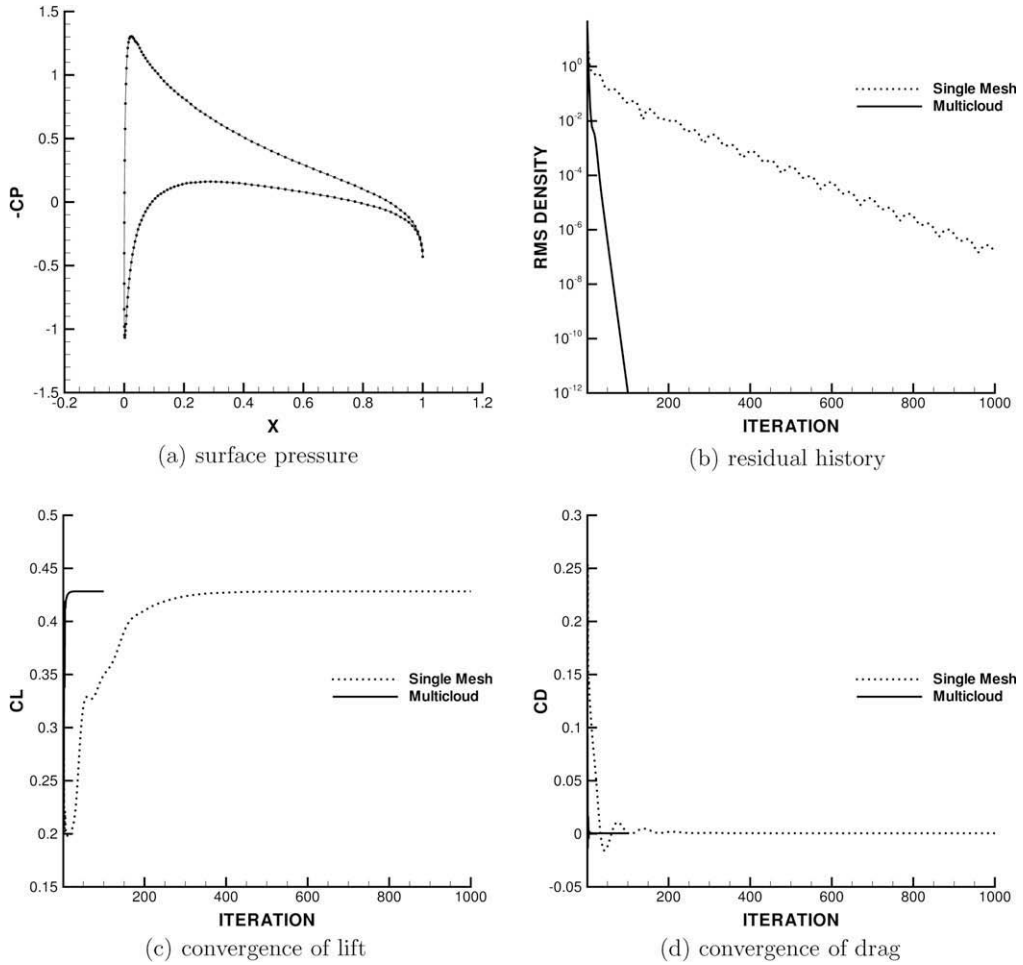


Fig. 8. Flow over NACA 0012, $M = 0.5$, $\alpha = 3.0^\circ$.

The diffusive flux, $\boldsymbol{d}_{k0}$, is also based on the CUSP scheme and is proportional to the metrics $\frac{1}{2}(x_{k+1} - x_{k-1})$ and $\frac{1}{2}(y_{k+1} - y_{k-1})$, as shown in Fig. 7(b).

The third fine level scheme tested with multicloud was a cell finite volume (CFV) scheme, as shown in Fig. 7(c). In the CFV scheme, the flow variables are stored at the cell centers, and the semi-discrete form of the Euler equations at cell 0 surrounded by neighboring cells k, may be expressed simply as

$$V_0 \frac{d\boldsymbol{w}_0}{dt} + \sum_k \boldsymbol{F}_{0k} = 0, \tag{24}$$

where the directed flux, $\boldsymbol{F}_{0k}$, is defined in Eq. (22). The artificial diffusion term is again based on the CUSP scheme. Reconstruction of left and right states is performed by first transferring the solution to the nodes with a conservative weighting procedure, as described by Jameson and Vassberg [22]. Subsequently, estimates of the change in the solution on either side of a given edge may be obtained from the nodal values opposite the edge.

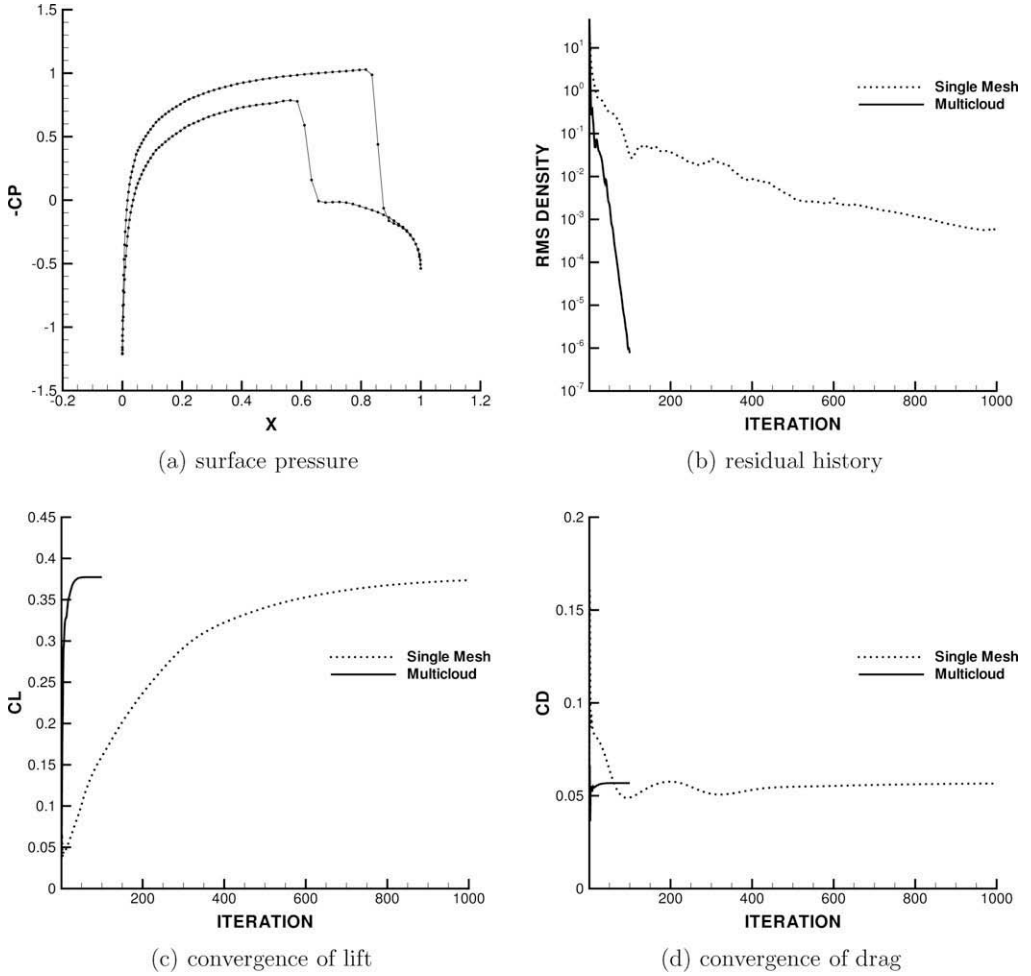
A structured finite volume (FV) scheme using structured multigrid, shown in Fig. 7(d), was used as a baseline for comparison. The FV scheme was implemented using the conservative numerical flux approach of Jameson [19,20], in which the semi-discrete form of the Euler equations at cell (i,j) may be expressed as

$$V_{ij} \frac{d\boldsymbol{w}_{ij}}{dt} + \boldsymbol{h}_{i+\frac{1}{2}j} - \boldsymbol{h}_{i-\frac{1}{2}j} + \boldsymbol{h}_{ij+\frac{1}{2}} - \boldsymbol{h}_{ij-\frac{1}{2}} = 0, \tag{25}$$

where the numerical flux is defined as

$$\boldsymbol{h}_{i+\frac{1}{2}j} = \frac{1}{2}(\boldsymbol{F}_{ij} + \boldsymbol{F}_{i+1j}) - \frac{1}{2}(\boldsymbol{d}_{ij+\frac{1}{2}}).$$

The artificial diffusion term, $\boldsymbol{d}_{ij+\frac{1}{2}}$, is again based on CUSP.



(a) surface pressure (b) residual history

(c) convergence of lift (d) convergence of drag

Fig. 9. Flow over NACA 0012, $M = 0.85$, $\alpha = 1.0^\circ$.

The dramatic effect of multicloud acceleration using four levels for the NFV scheme may be seen in Figs. 8–10. The figures show surface pressure, residual history, and convergence of both lift and drag for three separate test cases. For all cases tested, multicloud reduced the amount of time to attain steady state lift and drag within 1% by a factor of around 20. The performance of multicloud for these three cases was compared with the FV structured multigrid algorithm. The results of this comparison are shown in Table 1. As the table indicates, convergence rates per unit work with four levels of multicloud are comparable to those obtained with the FV scheme. Here, a work unit is the CPU time spent on the fine grid, assuming each node requires the same CPU time regardless of level. This is particularly promising, since the FV scheme is based on structured multigrid, which is a proven and well-established method [17]. It should be emphasized that since convergence rates are expressed per unit work, the effect of the non-ideal coarsening of Fig. 6 has been factored into the comparison, showing no adverse effects.

Finally, Table 2 highlights the fact that multicloud works well for all three fine level schemes described in this section. The HRM, NFV, and CFV schemes were tested using multicloud for the NACA 0012 at $M = 0.8$ and $\alpha = 1.25^\circ$. In addition, the CFV scheme was tested for the same case but using conventional non-nested multigrid, in which separate triangulations were generated independently. The structured FV multigrid scheme was also used as a baseline for comparison. As the table

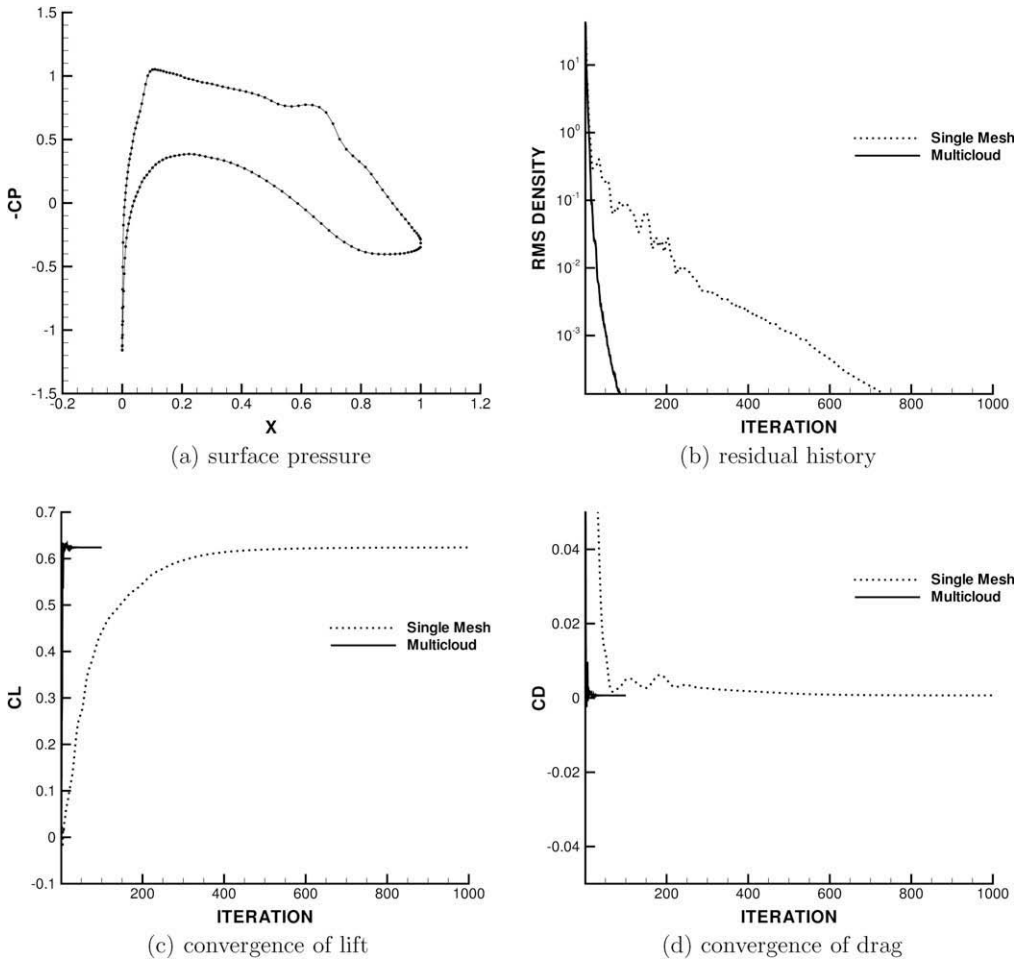


Fig. 10. Flow over KORN airfoil, $M = 0.75,\ \alpha = 0.0^\circ$.

Table 1
Comparison of convergence rate per unit work for the unstructured NFV scheme with multicloud versus the FV scheme with nested multigrid.

Case tested	Multicloud (NFV)		Multigrid (FV)	
	Single grid	Four-levels	Single grid	Four-levels
NACA 0012, $M = 0.5,\ \alpha = 3.0^\circ$	0.981	0.872	0.982	0.880
NACA 0012, $M = 0.85,\ \alpha = 1.0^\circ$	0.989	0.916	0.990	0.926
KORN, $M = 0.75,\ \alpha = 0.0^\circ$	0.985	0.945	0.987	0.940

Table 2
Comparison of convergence rate per unit work for various fine level schemes for the NACA 0012, $M = 0.8$, $\alpha = 1.25^{\circ}$.

Fine level scheme	Convergence accelerator	Single grid	Four-levels
HRM	Multicloud	0.983	0.921
NFV	Multicloud	0.986	0.922
CFV	Multicloud	0.990	0.955
CFV	Non-nested multigrid	0.992	0.959
FV	Nested multigrid	0.988	0.932

shows, multicloud was effective at accelerating convergence with all three schemes. The cell-centered scheme was significantly slower than the meshless or nodal schemes, but this is more likely a property of the scheme itself rather than multicloud, since the CFV scheme with non-nested multigrid showed similar behavior. Once again, convergence acceleration was as dramatic for multicloud as for the established structured multigrid observed with the FV scheme. One of the most notable features of multicloud is its transparent ability to accelerate convergence for a variety of fine level schemes. While algebraic methods offer similar mesh transparency, multicloud offers a geometric approach to achieve a similar goal.

6. Conclusions

In conclusion, multicloud represents a powerful convergence accelerator for iterative schemes, which appears to work equally well for a variety of fine level algorithms. The scheme transparency of multicloud will allow researchers to develop accurate schemes independent of any convergence accelerator, since multicloud is a versatile, modular algorithm. Since no assumptions are made regarding the topology of the fine level mesh, multicloud can be implemented on arbitrary mesh types in a mesh transparent manner. While coarsening procedures used in multicloud result in non-ideal coarsening ratios, this does not appear to degrade convergence when compared per unit work with established methods. The two-dimensional results presented here indicate the vast potential of multicloud to accelerate complex three-dimensional cases in the future.

Acknowledgments

This work was performed under the support of a National Defense Science and Engineering Graduate (NDSEG) Fellowship and the AFOSR. The first author has benefited greatly from the NDSEG program administered through the High Performance Computing Modernization Office of the Department of Defense. Likewise, the second author has benefited greatly from the long term and continuing support of the AFOSR Computational Mathematics Program directed by Dr. Fariba Fahroo.

References

[1] S.N. Atluri, T. Zhu, A new meshless local Petrov–Galerkin (mlpg) approach in computational mechanics, Computational Mechanics 22 (1998) 117–127.
[2] N. Balakrishnan, C. Praveen, A new upwind least squares finite difference scheme (lsfd-u) for Euler equations of gas dynamics, in: F. Benkhaldoun, R. Vilsmeier, D. Hänel (Eds.), Finite Volumes for Complex Applications, vol. II, Hermes Science Publications, 1999, p. 331.
[3] J.T. Batina, A gridless Euler/Navier–Stokes solution algorithm for complex aircraft applications, in: AIAA 31st Aerospace Sciences Meeting and Exhibit, AIAA Paper 1993-0333, Reno, NV, January 1993.
[4] T. Belytschko, Y.Y. Lu, L. Gu, Element-free Galerkin methods, International Journal for Numerical Methods in Engineering 37 (1994) 229–256.
[5] A. Brandt, Multi-level adaptive solutions to boundary-value problems, Mathematics of Computation 31 (1977) 333–390.
[6] T. Chan, B. Smith, Domain decomposition and multigrid algorithms for elliptic problems on unstructured meshes, Electronic Transactions on Numerical Analysis 2 (1994) 171–182.
[7] P.P. Chinchapatnam, K. Djidjeli, P.B. Nair, Meshless rbf collocation for steady incompressible viscous flows, in: AIAA 36th Fluid Dynamics Conference and Exhibit, AIAA Paper 2006-3525, San Francisco, CA, June 2006.
[8] P. Crumpton, M. Giles, Implicit time accurate solutions on unstructured dynamic grids, in: AIAA 12th Computational Fluid Dynamics Conference, AIAA Paper 1995-1671, San Diego, CA, June 1995.
[9] A. de Boer, M. van der Schoot, H. Bijl, Mesh deformation based on radial basis function interpolation, Computers and Structures 85 (2007) 784–795.
[10] E. Divo, A.J. Kassab, A meshless method for conjugate heat transfer problems, Engineering Analysis with Boundary Elements 29 (2005) 136–149.
[11] E. Divo, A.J. Kassab, Efficient localized meshless modeling of natural convective viscous flows, in: Ninth AIAA/ASME Joint Thermophysics and Heat Transfer Conference, AIAA Paper 2006-3089, San Francisco, CA, June 2006.
[12] C. Duarte, J. Oden, Hp clouds – a meshless method to solve boundary-value problems, Technical Report 95-05, Texas Institute for Computational and Applied Mathematics, Austin, TX, 1995.
[13] R. Fedorenko, The speed of convergence of one iterative process, USSR Computational Mathematics and Mathematical Physics 4 (1964) 227–235.
[14] R. Franke, Scattered data interpolation: tests of some methods, Mathematics of Computation 38 (1982) 181–200.
[15] F.C. Gunther, W.K. Liu, Implementation of boundary conditions for meshless methods, Computer Methods in Applied Mechanics and Engineering 163 (1998) 205–230.
[16] Y.C. Hon, R. Schaback, On unsymmetric collocation by radial basis functions, Applied Mathematics and Computation 119 (2001) 177–186.
[17] A. Jameson, Solution of the Euler equations for two-dimensional transonic flow by a multigrid method, Applied Mathematics and Computation 13 (1983) 327–355.
[18] A. Jameson, Multigrid algorithms for compressible flow calculations, in: W. Hackbusch, U. Trottenberg (Eds.), Lecture Notes in Mathematics, Springer-Verlag, 1986, pp. 166–201.
[19] A. Jameson, 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 4 (1995) 171–218.
[20] A. Jameson, Analysis and design of numerical schemes for gas dynamics. 2. Artificial diffusion and discrete shock structure, International Journal of Computational Fluid Dynamics 5 (1995) 1–38.
[21] A. Jameson, T.J. Baker, N.P. Weatherill, Calculation of inviscid transonic flow over a complete aircraft, in: AIAA 24th Aerospace Sciences Meeting, AIAA Paper 1986-0103, Reno, NV, January 1986.

[22] A. Jameson, J. Vassberg, A vertex–centroid (v–c) scheme for the gas-dynamics equations, Tech. Rep., International Conference on CFD, Kyoto, Japan, July 2000.
[23] E.J. Kansa, Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics—I, Computers and Mathematics with Applications 19 (1990) 127–145.
[24] E.J. Kansa, Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics—II, Computers and Mathematics with Applications 19 (1990) 147–161.
[25] A. Katz, A. Jameson, Edge-based meshless methods for compressible flow simulations, in: AIAA 46th Aerospace Sciences Meeting and Exhibit, Reno, AIAA Paper 2008-669, NV, January 2008.
[26] M. Lallemand, H. Steve, A. Dervieux, Unstructured multigridding by volume agglomeration: current status, Computers and Fluids 21 (1992) 397–433.
[27] J. Li, A.H.D. Cheng, C. Chen, A comparison of efficiency and error convergence of multiquadric collocation method and finite element method, Engineering Analysis with Boundary Elements 27 (2003) 251–257.
[28] G.R. Liu, Y.T. Gu, An Introduction to Meshfree Methods and Their Programming, Springer, 2005.
[29] J. Liu, S. Su, A potentially gridless solution method for the compressible Euler/Navier–Stokes equations, in: AIAA 34th Aerospace Sciences Meeting and Exhibit, AIAA Paper 1996-0526, Reno, NV, January 1996.
[30] W.K. Liu, S. Jun, Y.F. Zhang, Reproducing kernel particle methods, International Journal for Numerical Methods in Fluids 20 (1995) 1081–1106.
[31] R. Löhner, K. Morgan, Unstructured multigrid methods, Paper Presentation, Second European Conference on Multigrid Methods, October 1985.
[32] R. Löhner, C. Sacco, E. Oñate, S. Idelsohn, A finite point method for compressible flow, International Journal for Numerical Methods in Engineering 53 (2002) 1765–1779.
[33] D. Mavriplis, A. Jameson, Multigrid solution of the two-dimensional Euler equations on unstructured triangular meshes, in: AIAA 25th Aerospace Sciences Meeting, AIAA Paper 1987-0353, Reno, NV, January 1987.
[34] K. Morinishi, Effective accuracy and conservation consistency of gridless type solver, in: N. Satofuka (Ed.), Computational Fluid Dynamics 2000: Proceedings of the First International Conference on Computational Fluid Dynamics, Springer-Verlag, 2000, pp. 325–330.
[35] J. Müller, On Triangles and Flow, Ph.D. Thesis, The University of Michigan, 1996.
[36] R. Ni, A multiple grid scheme for solving the Euler equations, in: AIAA Fifth Computational Fluid Dynamics Conference, AIAA Paper 1981-1025, Palo Alto, CA, June 1981.
[37] E. Oñate, S. Idelsohn, A mesh-free finite point method for advective–diffusive transport and fluid flow problems, Computational Mechanics 21 (1998) 283–292.
[38] E. Oñate, S. Idelsohn, O.C. Zienkiewicz, R.L. Taylor, A finite point method in computational mechanics, applications to convective transport and fluid flow, International Journal for Numerical Methods in Engineering 39 (1996) 3839–3866.
[39] E. Oñate, S. Idelsohn, O.C. Zienkiewicz, R.L. Taylor, C. Sacco, A stabilized finite point method for analysis of fluid mechanics problems, Computer Methods in Applied Mechanics and Engineering 139 (1996) 315–346.
[40] E. Perez, Finite Element and Multigrid Solution of the Two-Dimensional Euler Equations on a Non-Structured Mesh, Report 442, INRIA, September 1985.
[41] H. Power, V. Barraco, Comparison analysis between unsymmetric and symmetric radial basis function collocation methods for the numerical solution of partial differential equations, Computers and Mathematics with Applications 43 (2002) 551–583.
[42] C. Praveen, N. Balakrishnan, New least squares based finite difference scheme for compressible flows, in: Proceedings of the Eighth Asian Congress of Fluid Mechanics, Tech. Rep., Shenzen, China, December 1999.
[43] J. Ruge, K. Stüben, Algebraic multigrid, in: S. McCormick (Ed.), Multigrid Methods, SIAM, 1987, pp. 73–130.
[44] B. Sarler, R. Vertnik, Meshfree explicit local radial basis function collocation method for diffusion problems, Computers and Mathematics with Applications 51 (2006) 1269–1282.
[45] R. Schaback, H. Wendlend, Characterization and construction of radial basis functions, in: N. Dyn, D. Leviatan, D. Levin, A. Pinkus (Eds.), Multivariate Approximations and Applications, Cambridge University Press, 2001, pp. 1–24.
[46] M. Sharan, E.J. Kansa, S. Gupta, Application of the multiquadric method for numerical solution of elliptic partial differential equations, Applied Mathematics and Computation 84 (1997) 275–302.
[47] C. Shu, H. Ding, H.Q. Chen, T.G. Wang, An upwind local rbf-dq method for simulation of inviscid compressible flows, Computer Methods in Applied Mechanics and Engineering 194 (2005) 2001–2017.
[48] D. Sridar, N. Balakrishnan, An upwind finite difference scheme for meshless solvers, Journal of Computational Physics 189 (2003) 1–29.
[49] K. Stüben, An introduction to algebraic multigrid, in: U. Trottenberg, C. Oosterlee, A. Schüller (Eds.), Multigrid, Academic Press, 2001, pp. 413–532.
[50] V. Venkatakrishnan, D.J. Mavriplis, Agglomeration multigrid for the three-dimensional Euler equations, in: AIAA 32nd Aerospace Sciences Meeting and Exhibit, Reno, AIAA Paper 1994-0069, NV, January 1994.
[51] J. Weiss, J. Maruszewski, W. Smith, Implicit solution of the Navier–Stokes equations on unstructured meshes, in: AIAA 13th Computational Fluid Dynamics Conference, AIAA Paper 1997-2103, Snowmass Village, CO, June 1997.
[52] H. Wendlend, Meshless Galerkin methods using radial basis functions, Mathematics of Computation 68 (1999) 1521–1531.