Multicloud Convergence Acceleration for Complex Applications on Arbitrary Grids

Aaron Katz* and Antony Jameson†
Department of Aeronautics and Astronautics, Stanford University, Stanford, CA
ajkatz@stanford.edu and jameson@baboon.stanford.edu

Abstract

A method is presented which dramatically enhances convergence of discrete systems for a wide variety of discretizations, including finite difference, finite volume, and mixed element approaches. The method, termed "multicloud," was first introduced in a previous work by Katz and Jameson.[5] This work concerns two major enhancements to multicloud: 1) The integration of multicloud procedures for arbitrary and mixed meshes, and 2) the development of automatic coarsening procedures to obtain coarse levels for the multicloud procedure. The first enhancement involves the use of meshless scheme to damp out unresolvable high frequency modes present on a conventional mesh. Because the meshless scheme is only used on coarse levels, the final solution retains any and all conservative properties of the parent mesh. The second enhancement allows for the automatic creation of coarse point distributions given a conventional mesh. Thus, the multicloud procedure may be used with no extra mesh generation tools needed. These two enhancements create a powerful convergence acceleration technique which is entirely mesh transparent, lending itself well to complex configurations in which mixed grid types are used and for which efficiency is of the utmost importance.

1. Introduction

The past two decades in computational aerodynamics have been dominated by vast improvements in algorithmic accuracy and efficiency. A core component in the improvements in efficiency has been the wide spread use of multigrid algorithms.[2] While first developed for structured meshes[1], multigrid has been extended to unstructured meshes with much success[7]. Whatever the mesh topology, the basic ideas behind multigrid remain the same: solve a given problem on a series of coarse discretizations and transfer the resulting corrections back to the fine mesh. The result of such a procedure is the accelerated damping of high frequency modes which are unresolvable on the finest mesh. The power of multigrid is a result of the ability to take large time steps on coarse grids combined with minimal overhead due to small numbers of points contained in coarse meshes.

Underlying the principles of multigrid is the need for coarse meshes which optimally complement the fine mesh. For structured meshes, the obvious solution is to eliminate every other node in each coordinate direction to obtain a coarse mesh. This procedure may be performed recursively to efficiently obtain a series of coarse meshes. This procedure does not directly extend to unstructured meshes, which are often used to automatically accommodate highly complex geometry. Instead, agglomeration procedures[9] are used which fuse fine cells into larger coarse cells. The agglomeration proceeds by an advancing front technique in which the faces along the front are composed of the latest agglomerated cells. The result of such procedures is a coarsened mesh consisting of large, multi-faceted control volumes. While many of the facets maybe combined, coarse mesh solvers must be able to accommodate arbitrary polyhedra. Though powerful, agglomeration requires sophisticated procedures to produce coarse meshes suitable for multigrid computations.

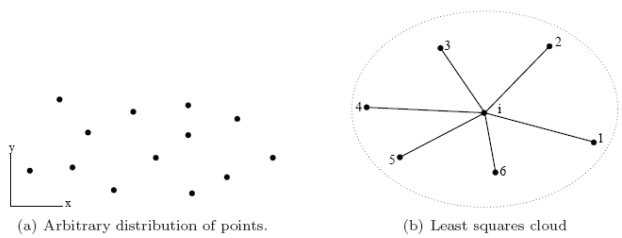
In this work, a meshless scheme is used on coarse levels in place of agglomerated finite volume methods. The use of the meshless scheme on coarse levels, termed "multicloud," allows for much greater flexibility and efficiency in the generation of coarse point distributions than agglomeration procedures allow. Coarsening procedures which produce suitable point distributions and connectivity are extremely simple and fast. The only inputs to the coarsening procedure are a list of coordinates and edges of the fine grid. Since all conventional mesh topologies may be expressed as lists of coordinates and edges, the multicloud procedure is entirely mesh transparent with the potential to impact structured, unstructured, prismatic, and mixed meshes. Multicloud may also be used with various discretization schemes. In

*PhD Candidate, Aeronautics and Astronautics, Stanford, CA.
†Thomas V. Jones, Professor of Engineering, Aeronautics and Astronautics, Stanford, CA

978-0-7695-3515-9/08 $25.00 © 2008 IEEE
DOI 10.1109/DoD.HPCMP.UGC.2008.64

this work, three schemes are tested with multicloud: 1) the meshless scheme of Katz and Jameson[5], 2) anode-centered finite volume scheme, and 3) a cell-centered finite volume scheme. In addition, since the meshless scheme is only used on the coarse levels, the fine mesh solution retains any and all of its own conservative properties.

This paper begins with a description of a simplified meshless scheme for coarse levels. Subsequently, the point coarsening procedure is discussed, along with a description of multicloud transfer operators. Finally, results are presented which show the effect of multicloud for two-dimensional (2D) inviscid flow problems.



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

Figure 1. Domain discretization for least squares derivatives

2. Meshless Scheme for Coarse Levels

The derivation of the meshless scheme used on coarse levels follows closely to the method described by Katz and Jameson.[5] Consider an arbitrary function $\phi(x,y)$ represented by a set of discrete nodal values in a 2D domain, as shown in Figure 1(a). Furthermore, consider node i surrounded by n nearby nodes as shown in Figure 1(b), forming a *cloud* nearest neighbors around i. The function ϕ may be expanded in a truncated Taylor series about ito all of its cloud points. With a sufficient number of points, an over determined system of equations for the derivatives of ϕ results. Derviatives to a specified order of accuracy maybe obtained as shown by Sridar.[8] For the sake of brevity, the details of the least squares problem may be found in a previous work by the same authors.[5] It suffices to say that the result of the least squares procedure is that the gradient of ϕ may be expressed as weighted sums over the points j in the cloud of i:

$$\nabla \phi_i \approx \sum_{j=1}^{n} \bar{a}_{ij} \Delta \phi_{ij}, \tag{1}$$

where $\Delta\phi_{ij}=\phi_j-\phi_i$, and the least squares coefficients, $\bar{a}_{ij}=(a_{ij}, b_{ij})$, contain only geometric information obtained from solving the above least squares problem in a preprocess step. Additionally, inverse distance weighting may be applied to the least squares problem to better condition the least squares matrix.[6]

The above least squares method for derivatives may be applied to the Euler equations in strong conservation law form:

$$\frac{\partial w}{\partial t} + \frac{\partial f}{\partial x} + \frac{\partial g}{\partial y} = 0, \tag{2}$$

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

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

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

Substituting the least squares framework directly into the spatial terms in Eq. 2 at a point i results in

$$\frac{\partial w_i}{\partial t} + \sum_{j=1}^{n} a_{ij} \Delta f_{ij} + \sum_{j=1}^{n} b_{ij} \Delta g_{ij} = 0, \tag{3}$$

or in terms of a directional flux, $F = af + bg$,

$$\frac{\partial w_i}{\partial t} + \sum_{j=1}^{n} \Delta F_{ij} = 0. \tag{4}$$

Eq. 4 represents a non-dissipative, unstable discretization. Stabilization maybe conveniently introduced via a numerical flux function defined for each edge ij connecting two meshless points. Using the midpoint flux at $j+\frac{1}{2}$ leads to the following discretization of the spatial derivatives,

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

where the factor of two is needed since the expansion is only taken to the midpoint of the edge. A general framework for stable schemes is obtained by averaging the endpoint fluxes, augmented with diffusive terms:

$$F_{j+\frac{1}{2}} = \frac{1}{2}\left(F_i + F_j\right) - \frac{1}{2} d_{j+\frac{1}{2}} \tag{5}$$

Since the meshless scheme is only used on coarse levels, first order scalar dissipation may be used to save computational effort:

$$d_{j+\frac{1}{2}} = \alpha_{ij}\left(w_j - w_i\right), \tag{6}$$

where α_{ij} is proportional to the spectral radius of the directed flux Jacobian for the edge.

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

$$\frac{\partial w_i}{\partial t} + R_i = 0, \tag{7}$$

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 treated separately for use with the Runge-Kutta schemes for steady state problems derived by Jameson.[3]

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

$$\Delta t_i = \frac{CFL}{\sum_{j=1}^{n} \left(\left| a_{ij} u + b_{ij} v \right| + \sqrt{a_{ij}^2 + b_{ij}^2} c \right)}, \tag{8}$$

where u, v, and c are the local velocities and speed of sound. Along with local time stepping, implicit residual smoothing[4] was used to accelerate convergence. Implicit residual smoothing was implemented in a manner similar to that of edge-based unstructured grid algorithms:

$$\overline{R}_i = R_i + \epsilon \nabla^2 \overline{R}_i \tag{9}$$

The solution of Eq. 9 is costly to obtain in general, but two Jacobi iterations were sufficient to increase the allowable CFL by a factor of two.

3. Point Coarsening Procedure

The power of the above meshless scheme hinges on the ability to automatically coarsen a given mesh to obtain a suitable point distribution. In the context of the meshless scheme, a suitable point distribution means obtaining points and edges which produce well conditioned least squares problems and accurate derivatives. Thus the coarsening process is two-fold: 1) From a fine level, obtain a subset of points which represents the coarse point distribution, and 2) connect this subset of points to form edges which implicitly define a local cloud for each point.

The first step of the coarsening procedure is illustrated in Figure 2(a). 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, its nearest neighbors are blanked. The next point is then tested, until all the points are 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.

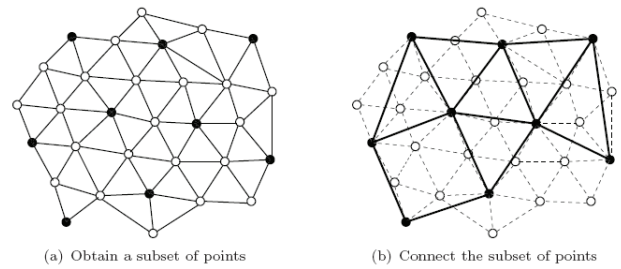


Figure 2. Point coarsening procedure

The second step in the coarsening procedure, illustrated in Figure 2(b), is to connect the valid coarse level points. For each blanked point, the valid nearest neighbors are connected to form an edge. It is possible that the blanking procedure performed in the first step could result in a single valid point among the nearest neighbors of a blanked point. In this case, the status of the blanked point is changed to "valid," and an edge between this new valid point and the single valid nearest neighbor is added.

The above coarsening procedure is extremely simple and fast. It may be applied to any mesh topology, since all conventional meshes may be expressed in terms of points interconnected with edges. Moreover, the procedure may be invoked recursively to form a series of coarse levels. So far, the procedure has shown to be robust, producing good quality point distributions up to five levels deep. Results from the procedure for four levels are shown in Figure 4.

While the finest level may consist of a mesh, it should be emphasized that the coarse levels are not meshes, but clouds of points defined implicity by edges. An edge in the meshless sense is simply two neighboring nodes that belong to the local clouds of one another. A list consisting of local clouds of neighboring nodes for each node in the domain completes the connectivity for meshless schemes. In this work, the list of local clouds is expressed as an edge list for ease of implementation. Note that edges often cross during the coarsening procedure. This presents no difficulty since the coarse level algorithm is entirely meshless.

4. Multicloud Transfers

Of course, suitable coarse levels upon which to apply the meshless method is only part of the multicloud framework. Additionally, transfer 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 set of transfer operators for all levels. The only difference present on the fine mesh is in the case of a cell-centered scheme. For a cell centered scheme, residuals and solution variables are first transferred to the nodes by a conservative volume weighting procedure. The multicloud transfers are invoked as usual. Finally, the nodal corrections obtained on the fine grid are interpolated back to the cell centers, completing the process for a cell-centered scheme.

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

$$\mathbf{w}_{\mathbf{k}}^{(0)} = T_{k,k-1}\mathbf{w}_{\mathbf{k}-1}. \tag{10}$$

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

$$\mathbf{P}_{\mathbf{k}} = Q_{k,k-1}\mathbf{R}_{\mathbf{k}-1}\left(\mathbf{w}_{\mathbf{k}-1}\right) - \mathbf{R}_{\mathbf{k}}\left(\mathbf{w}_{\mathbf{k}}^{(0)}\right). \tag{11}$$

The forcing function, $\mathbf{P_k}$, represents the difference between the aggregated fine cloud residuals, and the residuals computed with the coarse cloud solution. Subsequently, $\mathbf{R_k}(\mathbf{w_k})$ is replaced by $\mathbf{R_k}(\mathbf{w_k})+\mathbf{P_k}$ in the 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, w_k^+. Coarse level corrections, based on the difference between the corrected solution and the original solution transferred from the fine level, are then transferred back to the fine grid with an interpolation-like operator, $I_{k-1,k}$,

$$\mathbf{w}_{\mathbf{k}-1}^{+} = \mathbf{w}_{\mathbf{k}-1} + I_{k-1,k}\left(\mathbf{w}_{\mathbf{k}}^{+} - \mathbf{w}_{\mathbf{k}}^{(0)}\right). \tag{12}$$

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, illustrated in Figure 3 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. The injection process is shown in Figure 3(a).

The other two operators involve coincident and non-coincident points. It can be seen from Figure 3 that all fine level points are either coincident with a coarse level point, or have at least two coarse level points as nearest neighbors. Because the figure shows only a mesh patch, some nodes on the boundary of this patch may appear to have only one valid nearest neighbor. If the entire mesh could be shown, two valid nearest neighbors would always be present. This is enforced by the coarsening procedure itself. For a fine level point, i, with coincident or nearest neighbor coarse level points, j, an inverse distance weight coefficient may be defined:

$$c_{ij} = \frac{1}{\left(\Delta x_{ij}^2 + \Delta y_{ij}^2\right)^{\frac{g}{2}}} \qquad \textit{if not coincident}, \tag{13}$$

$$c_{ij}=1 \qquad \textit{if coincident}. \tag{14}$$

Both the residual transfer operator and the correction transfer operator rely on the distance weights of Eq. 13. The residual transfer operator involves distributing the residual at each fine level point to the valid nearest neighbor coarse level points with a weighting, as shown in Figure 3(b). The amount of residual given from fine level point i to coarse level point j is

$$\frac{c_{ij}R_i}{\sum_j c_{ij}}\left(\frac{ds_i}{ds_j}\right)^d, \tag{15}$$

where ds is the average edge length in the cloud at a 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.

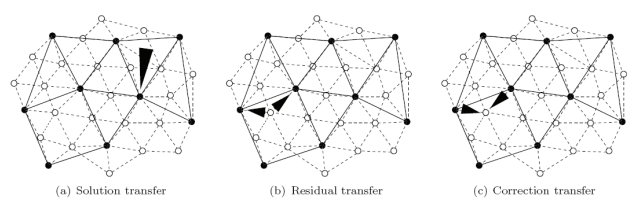


Figure 3. Summary of multigrid transfer operators

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

$$w_i^+ = w_i + \frac{\sum_j c_{ij}\left(w_j^+ - w_j^{(0)}\right)}{\sum_j c_{ij}}. \tag{16}$$

Thus all operators are either injection or 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. 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.

5. Results and Future Work

Results of both enhancements to multicloud discussed in this paper are given. Results of the coarsening procedure are displayed in Figure 4. The parent unstructured mesh around the NACA 0012 airfoil is shown, followed by three coarse levels of meshless clouds. It can be seen that coarsening ratios remain around 3.0 for the four levels of coarsening shown. Also, edge to node ratios, remain nearly constant for all levels of coarsening. This appears to be an important factor in creating coarse levels which obtain maximum convergence.

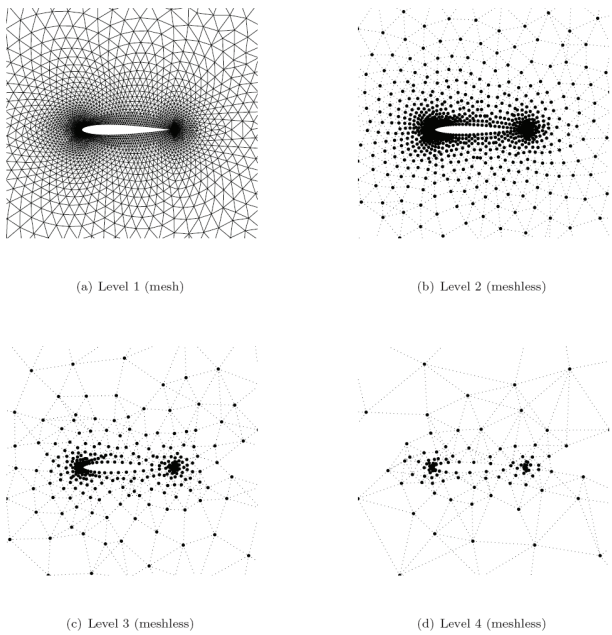


Figure 4. Level coarsening for NACA 0012 airfoil

Results showing the ability of multicloud to accelerate convergence for a variety of fine level discretization types, is shown in Figure 5. The top row of figures shows convergence for a high-order meshless scheme, while the middle and bottom rows show convergence for a node-centered finite volume scheme and a cell-centered finite volume scheme, respectively. It can be seen from these results, that multicloud is quite effective in accelerating convergence for the three schemes tested. Table 1 shows nearly an order of magnitude speed up for all three schemes with the use of four levels of multicloud. The results in Table 1 are based a definition of convergence dealing with the integrated global properties of lift and drag. A solution was considered converged if the global quantities obtained the steady state value to within 1.0%. The speed up in convergence of these integrated quantities was more dramatic than the speed up in residual.

Table 1. Level coarsening summary

Levels	Nodes	Edges	Edge:node ration	Nodal coarsening ratio
1	4,848	14,336	2.96:1	—
2	1,551	4,442	2.86:1	3.13
3	510	1,467	2.88:1	3.04
4	175	526	3.01:1	2.91

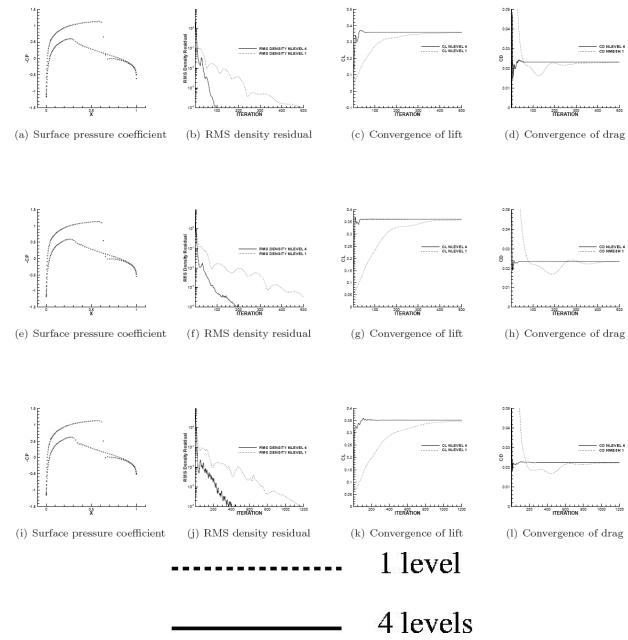


Figure 5. Effect of multicloud on convergence for NACA 0012, M=0.80, α=1.25° with various fine level schemes. Schemes plotted are meshless (top), node-centered finite volume (middle), cell-centered finite volume (bottom).

Results showing the speed up afforded by multicloud using the node-centered scheme on the fine level for a variety of test cases are shown in Figure 6. All coarse levels use the meshless scheme discussed in this work. Convergence of the global quantities of lift and drag are summarized in Table 3 for four test cases. The same convergence definition used in Table 2 was also used in Table 3. While the effects of multicloud are somewhat case dependent, all cases showed a speed up of between 5 and 8 times over a single level.

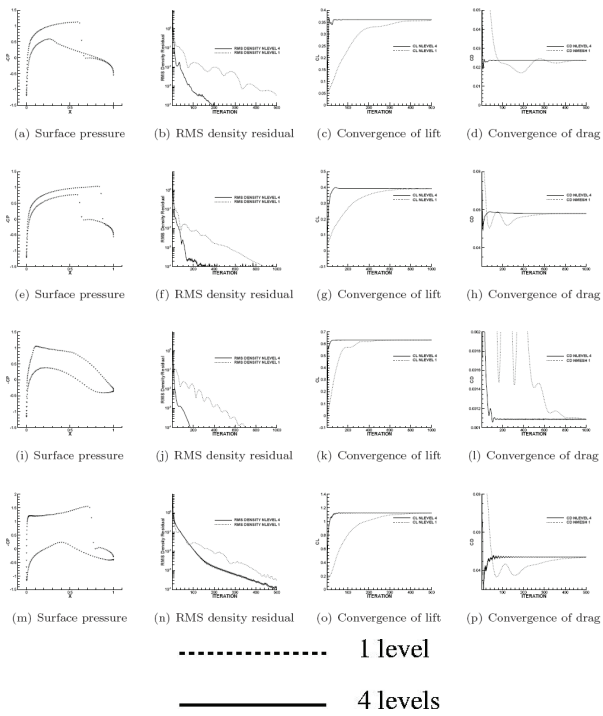


Figure 6. Effect of multicloud on convergence for the node-centered finite volume scheme on four test cases. Test cases include: NACA 0012, *M*=0.80, α=1.25° (top), NACA 0012, *M*=0.85, α=1.0° (second from top), KORN, *M*=0.75, α=0.0° (second from bottom), RAE 2822, *M*=0.75, α= 3.0° (bottom).

Table 2. Speed-up in CPU time for various fine level schemes for NACA 0012, *M*=0.80, α=1.25°

Levels	Meshless	Node-centered FV	Cell-centered FV
1	1.00	1.00	1.00
2	1.76	2.20	3.45
3	4.68	5.18	8.36
4	9.28	8.36	8.53

Table 3. Speed-up in CPU time for various flow cases using 4-level multicloud on a node-centered finite volume scheme

	Multicloud cycles		CPU time	
Levels	cycles (1/4)	speed-up	time (1/4)	speed-up
NACA 0012, *M*=0.80, α=1.25°	477/33	14.45	10.53/1.26	8.36
NACA 0012, *M*=0.85, α=1.0°	766/60	12.77	17.10/2.30	7.43
KORN, *M*=0.75, α=0.0°	928/98	9.47	24.33/4.42	5.50
RAE 2822, *M*=0.75, α=3.0°	452/43	10.51	12.02/1.96	6.13

These initial results are quite promising and prompt further investigation. Future work will focus on improving the convergence rate through two main avenues:

1. Obtain smoother coarse node distributions and point clouds, and
2. Modifying transfer operators.

Other research goals include testing the robustness of the coarsening procedure on a variety of test cases, and performing direct comparisons of multicloud versus agglomeration multigrid in terms of convergence acceleration.

Acknowledgements

Finally, this work was performed under the support of a National Defense Science and Engineering Graduate (NDSEG) Fellowship and the Air Force Office of Scientific Research (AFOSR). The first author has benefited greatly from the NDSEG program administered through the High Performance Computing Modernization Program 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. Jameson, A., "Solution of the euler equations for two dimensional transonic flow by a multigrid method." *Applied Mathematics and Computation*, 13, pp. 327–356, 1983.

2. Jameson, A., Multigrid algorithms for compressible flow calculations, in W. Hackbusch and U. Trottenberg, editors, *Lecture Notes in Mathematics*, Springer-Verlag, pp. 166–201, 1986.

3. 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*, 4, pp. 171–218, 1995.

4. Jameson, A. and D. Mavriplis, "Finite volume solution of the two-dimensional euler equations on a regular triangular mesh. AIAA Journal, 24, pp. 611–618, 1986.

5. Katz, A. and A. Jameson. Edge-based meshless methods for compressible flow simulations." *AIAA paper 2008-699*, AIAA 46th Aerospace Sciences Meeting, Reno, NV, January 2008.

6. 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.

7. Mavriplis, D. and A. Jameson, "Multigrid solution of two-dimensional euler equations on unstructured triangular meshes." *AIAA paper 87-0353*, AIAA 25th Aerospace Sciences Meeting, Reno, NV, January 1987.

8. Sridar, M. and N. Balakrishnan, "An upwind finite difference scheme for meshless solvers." *Journal of Computational Physics*, 189, pp. 1–29, 2003.

9. Venkatakrishnan, V. and D. Mavriplis, "Agglomeration multigrid for three-dimensional euler equations." *AIAA Journal*, 33, pp. 633–640, 1995.