17th AIAA Computational Fluid Dynamics Conference
6 - 9 June 2005, Toronto, Ontario Canada

AIAA 2005-5333

Parallel Implicit Adaptive Mesh Refinement Scheme for Body-Fitted Multi-Block Mesh

C. P. T. Groth* and S. A. Northrup†

University of Toronto Institute for Aerospace Studies
4925 Dufferin Street, Toronto, Ontario, M3H 5T6, Canada

A parallel implicit adaptive mesh refinement (AMR) algorithm is described for the system of partial-differential equations governing steady two-dimensional compressible gaseous flows. The AMR algorithm uses an upwind finite-volume spatial discretization procedure in conjunction with limited linear solution reconstruction and Riemann-solver based flux functions to solve the governing equations on multi-block mesh composed of structured curvilinear blocks with quadrilateral computational cells. A flexible block-based hierarchical data structure is used to facilitate automatic solution-directed mesh adaptation according to physics-based refinement criteria. A matrix-free inexact Newton method is used to solve the system of nonlinear equations arising from this finite-volume spatial discretization procedure and a preconditioned generalized minimal residual (GMRES) method is used to solve the resulting non-symmetric system of linear equations at each step of the Newton algorithm. Right preconditioning of the linear system is used to improve performance of the Krylov subspace method. An additive Schwarz global preconditioner with variable overlap is used in conjunction with block-fill incomplete lower-upper (BFILU) type preconditioners based on the Jacobian of the first-order upwind scheme for each sub-domain. The Schwarz preconditioning and block-based data structure readily allow efficient and scalable parallel implementations of the implicit AMR approach on distributed-memory multi-processor architectures. Numerical results are described for several flow cases, demonstrating both the effectiveness of the mesh adaptation and algorithm parallel performance. The proposed parallel implicit AMR method allows for anisotropic mesh refinement and appears to be well suited for predicting complex flows with disparate spatial and temporal scales in a reliable and efficient fashion.

I. Introduction

In spite of the relative maturity and widespread successes of computational fluid dynamics (CFD), there remain a variety of physically complex flows, which are still not well understood and which have proved to be very challenging to predict by numerical methods. Such flows would include but are not limited to: 1) turbulent flows in turbomachinery; 2) reactive flows associated with combustion processes; 3) compressible flows of conducting fluids and plasmas; and 4) transition-regime non-equilibrium flows. These flows are particularly challenging as they involve a wide range of complicated physical phenomena and several numerical challenges must be overcome in order to routinely solve such flows for both fundamental scientific research and engineering design and development. They are:

i) The prediction of physically complex flows places heavy demands on computational resources, requiring the solution of often large nonlinear systems of partial differential equations (PDEs).

*Associate Professor, Email: groth@utias.utoronto.ca, Senior Member AIAA
†PhD Candidate, Email: northrup@utias.utoronto.ca, Student Member AIAA

ii) Mathematical descriptions of these flows include a variety of complex physical processes, each with their own characteristic spatial and/or temporal scales. In many cases, the flows exhibit large disparities in the characteristic scales and solution techniques must deal with numerical stiffness arising from these disparities. For time-accurate calculations, allowable time steps are often dictated by stability constraints and not by the characteristic frequencies of the relevant physics and, for the solution of time-invariant problems, conventional schemes are generally non-optimal and convergence rates deteriorate with increasing mesh refinement.

iii) Calculations of physically complex flows require robust schemes that preserve positivity and realizability of a variety of solution variables.

Computational grids that automatically adapt to the solution of the governing PDEs are very effective in treating problems with disparate length scales, providing the required spatial resolution while minimizing memory and storage requirements. The use of adaptive mesh refinement (AMR) in conjunction with higher-order variants of Godunov-type upwind finite-volume schemes has led to some very powerful and robust methods for the treatment of flows with disparate length scales and complex and/or moving geometries. In particular, AMR approaches for Cartesian mesh[1–4] have proved to be very successful. Moreover, block-based variants of these techniques have been proposed for parallel implementations,[5–8] and other studies have considered the extensions to more arbitrary body-fitted mesh.[9–14]

Krylov subspace methods are iterative techniques for the solution of large sparse linear systems that when used in conjunction with Newton's method have proved to be very effective in the solution of nonlinear PDEs. Implicit methods based on the Newton-Krylov approach have gained widespread acceptance and would seem very appropriate for dealing with the problem of numerical stiffness arising from disparate temporal scales. In particular, the generalized minimal residual (GMRES) algorithm proposed by Saad and co-workers[15–18] for large sparse non-symmetric systems of linear equations has been successfully applied to the prediction of aerodynamic flows.[19–23] In addition, as indicated in the recent survey by Knoll and Keyes,[24] Newton-Krylov methods have been applied to a wide range of other physical problems. Nevertheless, effective parallel implementations of implicit Newton-Krylov methods are needed for performing practical scientific and engineering computations of physically complex flows.

One possible approach to the parallelization of Newton-Krylov methods is offered by domain-based additive Schwarz preconditioning techniques.[18] Keyes and co-researchers have achieved some impressive results in parallel implementations of Newton-Krylov-Schwarz (NKS) algorithms for the transonic full potential equations, low-Mach-number compressible combusting flows, and three-dimensional inviscid flows.[25–28] Furthermore, the Schwarz preconditioning technique would seem very compatible with multi-block AMR methods.[12–14]

This study focuses on the development of an effective solution strategy for physically complex steady flows with disparate spatial and temporal scales. A parallel implicit method is proposed for multi-block body-fitted mesh which combines a block-based AMR approach with a NKS solution of the discretized nonlinear equations governing compressible flows. The proposed AMR algorithm allows for anisotropic refinement on body-fitted meshes. Future research will focus on the further development of the algorithm for unsteady and three-dimensional flows and for more complex systems of equations.

II. Equations of Compressible Gas Dynamics

For the development of the proposed parallel implicit AMR algorithm, solutions of the equations governing compressible flows of polytropic gases are considered. For two-dimensional planar flows, the conservative form of the Euler equations of compressible gas dynamics reflecting the conservation of mass, momentum, and energy can be summarized as follows:

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = 0 \tag{1}$$

where $\mathbf{U}$ is the conserved variable solution vector given by

$$\mathbf{U} = \left[\begin{array}{cccc} \rho, & \rho u, & \rho v, & \rho e \end{array} \right]^T , \tag{2}$$

x and y are the spatial coordinates, t is time, ρ is the gas density, u and v are the velocity components in the x- and y-coordinate directions, $e = p/(\rho(\gamma - 1)) + (u^2 + v^2)/2$ is the specific total energy, $p = \rho RT$ is the pressure, T is the gas temperature, R is the gas constant, γ is the specific heat ratio, and $\mathbf{F}$ and $\mathbf{G}$ are x- and y-direction solution flux vectors given by

$$\mathbf{F} = \left[\begin{array}{c} \rho u \\ \rho u^2 + p \\ \rho uv \\ u\big(\rho e + p\big) \end{array} \right], \ \mathbf{G} = \left[\begin{array}{c} \rho v \\ \rho uv \\ \rho v^2 + p \\ v\big(\rho e + p\big) \end{array} \right]. \tag{3}$$

For a polytropic gas (thermally and calorically perfect gas), the ratio of specific heats, γ, is a constant and the specific heats are given by $C_v = R/(\gamma - 1)$ and $C_p = \gamma R/(\gamma - 1)$.

III. Parallel Implicit AMR Algorithm

A. Finite-Volume Scheme

The proposed AMR algorithm uses an upwind finite-volume spatial discretization procedure in conjunction with limited linear solution reconstruction and Riemann-solver based flux functions to solve the preceding equations on multi-block mesh composed of quadrilateral computational cells. The semi-discrete form of this finite-volume formulation applied to cell (i, j) is given by

$$\frac{d\mathbf{U}_{i,j}}{dt} = -\frac{1}{A_{i,j}} \sum_k \Big(\vec{\mathbf{F}} \cdot \vec{\mathbf{n}} \, \Delta\ell \Big)_{i,j,k} = \mathbf{R}_{i,j}(\mathbf{U}) \, , \tag{4}$$

where $\mathbf{U}_{i,j}$ is the conserved solution state for cell (i, j), $\vec{\mathbf{F}} = (\mathbf{F}, \mathbf{G})$ is the flux dyad, A_{ij} is the area of the cell, and $\Delta\ell$ and $\vec{\mathbf{n}}$ are the length of the cell face and unit vector normal to the cell face or edge, respectively. The vector, $\mathbf{R}$, is referred to as the residual vector. The numerical fluxes at the faces, k, of each cell, $\vec{\mathbf{F}} \cdot \vec{\mathbf{n}}$, are determined from the solution of a Riemann problem. Given the left and right solution states, $\mathbf{U}_l$ and $\mathbf{U}_r$, at the cell interfaces, the numerical flux is given by

$$\vec{\mathbf{F}} \cdot \vec{\mathbf{n}} = \mathcal{F}(\mathbf{U}_l, \mathbf{U}_r, \vec{\mathbf{n}}) \, , \tag{5}$$

where the numerical flux $\mathcal{F}$ is evaluated by solving a Riemann problem in a direction defined by the normal to the face with initial data $\mathbf{U}_l$ and $\mathbf{U}_r$. The left and right solution states are determined via a least-squares piece-wise limited linear solution reconstruction procedure in conjunction with either the Barth-Jesperson or Venkatakrishnan limiters.[29,30] In the present algorithm, both exact and approximate Riemann solvers can be used to solve the Riemann problem and evaluate the numerical flux. The Roe linearized Riemann solver,[31] HLLE and modified HLLE flux function due to Linde,[32–34] the HLLC flux function,[35] and the exact Riemann solver of Gottlieb and Groth[36] have all been implemented and may be used.

B. Block-Based Adaptive Mesh Refinement

Following the approach developed by Groth *et al.* for computational magnetohydrodynamics,[7,8] a flexible block-based hierarchical data structure has been developed and is used in conjunction with the finite-volume scheme described above to facilitate automatic solution-directed mesh adaptation on multi-block body-fitted quadrilateral mesh according to physics-based refinement criteria. The approach readily permits

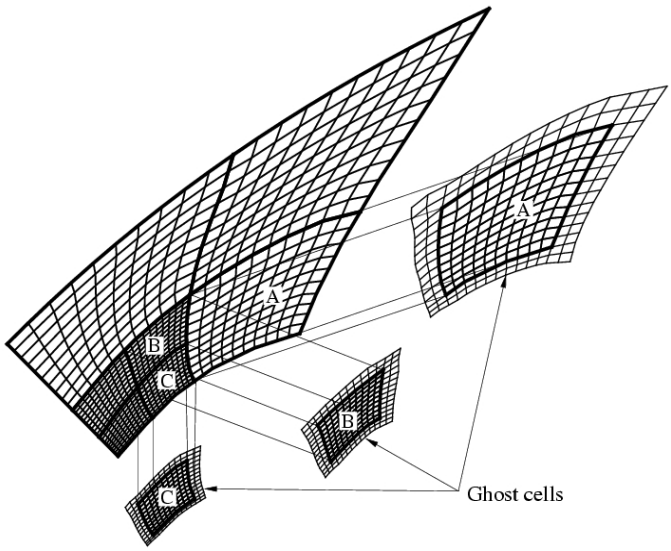


Figure 1. Multi-block body-fitted quadrilateral mesh of block-based AMR algorithm illustrating the layers of overlapping ghost cells used to facilitate inter-block communication.

local refinement of the mesh for complex flow geometries and also allows for anisotropic mesh refinement to resolve thin solution layers, such as boundary and free shear layers. The proposed AMR formulation borrows from previous work by Berger and co-workers,[1,3,6,37] Quirk,[5,38] and De Zeeuw and Powell[2] for Cartesian mesh and has similarities with the block-based approaches described by Quirk and Hanebutte[5] and Berger and Saltzman.[6] Note that other researchers have considered the extension of Cartesian mesh adaptation procedures to more arbitrary quadrilateral and hexagonal mesh. See for example the work by Davis and Dannenhoffer[9] and Sun and Takayama.[10] Note that although the proposed block-based AMR approach is somewhat less flexible and incurs some inefficiencies in solution resolution as compared to a cell-based approaches (i.e., for the same solution accuracy, generally more computational cells are introduced in the adapted grid), the block-based method can offer many advantages over cell-based techniques when computational performance and parallel implementation of the solution algorithm is considered.

In this work, the solution of the conservation equations by the finite-volume method of the preceding section provides area-averaged solution quantities within quadrilateral computational cells and these cells are embedded in structured blocks consisting of $N_{\text{cells}} = N_x \times N_y$ cells, where N_x and N_y are even, but not necessarily equal integers. Refer to Figure 1. Solution data associated with each block are stored in indexed array data structures and it is therefore straightforward to obtain solution information from neighbouring cells within blocks. Mesh adaptation is accomplished by the dividing and coarsening of appropriate solution blocks. In regions requiring increased cell resolution, a "parent" block is refined by dividing itself into four "children" or "offspring". Each of the four quadrants or sectors of a parent block becomes a new block having the same number of cells as the parent and thereby doubling the cell resolution in the region of interest. This process can be reversed in regions that are deemed over-resolved and four children are coarsened into a single parent block. To ensure that grid lines of the body fitted mesh do not merge and/or cross, an elliptic mesh smoothing algorithm[39,40] is applied to the newly created blocks of the computational mesh after refinement and coarsening.

Figures 1 and 2 illustrate the refinement and coarsening of the blocks. The mesh refinement is constrained such that the grid resolution changes by only a factor of two between adjacent blocks and the minimum resolution is not less than that of the initial mesh. Standard multigrid-type restriction and prolongation operators are used to evaluate the solution on all blocks created by the coarsening and division processes,

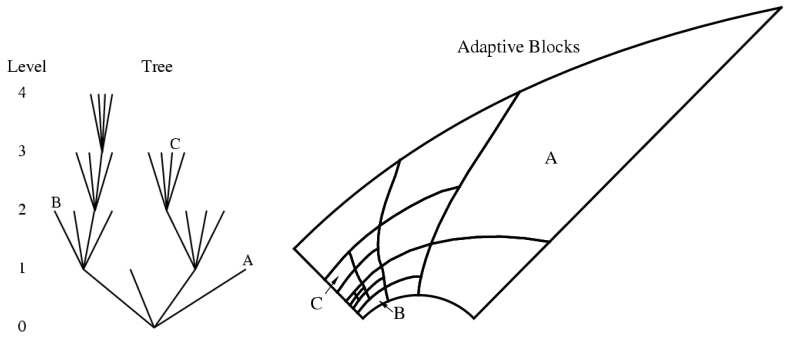


Figure 2. Solution blocks of a computational mesh with four refinement levels originating from one initial block and the associated hierarchical quadtree data structure. Interconnects to neighbours are not shown.

respectively. Although several approaches are possible, for this study, the coarsening and division of blocks are directed using multiple physics-based refinement criteria.[41] Refinement criteria based on a combination of the density gradient, and divergence and curl of the velocity provide reliable detection of flow features such as shocks, contact surfaces, stagnation points, and shear layers.

In order that the finite-volume scheme can be applied to all blocks in a more independent manner, some solution information is shared between adjacent blocks having common interfaces. This information is stored in an additional two layers of overlapping "ghost" cells associated with each block as shown in Figure 1. At interfaces between blocks of equal resolution, these ghost cells are simply assigned the solution values associated with the appropriate interior cells of the adjacent blocks. At resolution changes, restriction and prolongation operators, similar to those used in block coarsening and division, are employed to evaluate the ghost cell solution values. Within the AMR approach, additional inter-block communication is also required at interfaces with resolution changes to strictly enforce the flux conservation properties of the finite-volume scheme.[1,37] In particular, the interface fluxes computed on more refined blocks are used to correct the interface fluxes computed on coarser neighbouring blocks and ensure the solution fluxes are conserved across block interfaces.

A hierarchical tree-like data structure with multiple "roots", multiple "trees", and additional interconnects between the "leaves" of the trees is used to keep track of mesh refinement and the connectivity between solution blocks. This interconnected "forest" data structure is depicted in Figure 2. The blocks of the initial mesh are the roots of the forest which are stored in an indexed array data structure. Associated with each root is a separate "quadtree" data structure that contains all of the blocks making up the leaves of the tree created from the original parent blocks during mesh refinement. Each grid block corresponds to a node of the tree. Traversal of the multi-tree structure by recursively visiting the parents and children of solution blocks can be used to determine block connectivity. However, in order to reduce overhead associated with accessing solution information from adjacent blocks, the neighbours of each block are computed and stored, providing direct interconnects between blocks in the hierarchical data structure that are neighbours in physical space. One of the advantages of the hierarchical quadtree data structure is that it readily permits local mesh refinement. Local modifications to the multi-block mesh can be performed without re-gridding the entire mesh and re-calculating all solution block connectivities.

An example illustrating the adaptation of a multi-block quadrilateral mesh for a NACA 0012 aerofoil is shown in Figure 3. The figure shows a refined mesh derived from an initial C-type grid consisting of four blocks and 2,048 cells (10×32 and 22×32 cell blocks were used) by applying three levels of refinement. The resulting refined mesh consists of 22 blocks and 14,720 cells. The solution block boundaries (thick lines) and computational cells (thin lines) for both mesh are depicted in the figure. Note that each level of refinement in the grid introduces cells that are typically smaller by a factor two in each spatial dimension.

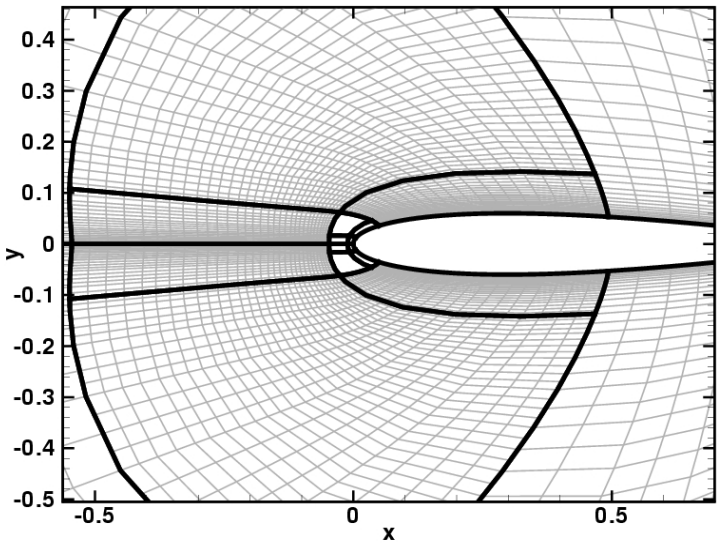


Figure 3. Illustration of AMR for two-dimensional multi-block quadrilateral mesh for NACA 0012 aerofoil. Final 22-block (14,720 cells) refined mesh obtained from 4-block initial mesh (2,048 cells) after three levels of refinement.

Practical calculations may have 10-15 levels of refinement. In the case of 15 levels of refinement, the finest cells in the mesh are more than 32,000 (2^{15}) times smaller than the coarsest cells. Note also that the initial mesh stretching, in this case applied to the mesh in a direction normal to the aerofoil surface and near the leading and trailing edges, is retained by the mesh refinement procedure such that the refined blocks and the cells within them are clustered near the aerofoil surface, with additional clustering near the leading and trailing edges. Use of cell stretching and clustering in the initial mesh enables anisotropic refinement of the multi-block grid, which will be particularly important for resolving boundary and shear layers in subsequent studies of viscous flows.

C. Inexact Newton Method

The semi-discrete form of the governing equations given in Eq. (4) form a coupled set of non-linear ordinary differential equations. In this work, steady-state solutions of these are of prime interest. For steady flows, time-invariant solutions Eq. (4) are sought satisfying

$$\mathbf{R}(\mathbf{U}) = 0\,, \tag{6}$$

requiring the solution of a large coupled nonlinear system of algebraic equations. Newton's method is a common, robust, and efficient iterative technique for the solution of nonlinear systems of this type and is used here. Starting with an initial estimate, $\mathbf{U}^0$, successively improved estimates for the solution are obtained by solving

$$\left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^n \Delta\mathbf{U}^n = \mathbf{J}^n \Delta\mathbf{U}^n = -\mathbf{R}(\mathbf{U}^n)\,, \tag{7}$$

at each step, n, of the Newton method, and an improved approximation for the solution is given by

$$\mathbf{U}^{n+1} = \mathbf{U}^n + \Delta\mathbf{U}^n\,. \tag{8}$$

Here $\mathbf{J} = \partial\mathbf{R}/\partial\mathbf{U}$ is the residual Jacobian. The iterative procedure is repeated until an appropriate norm of the solution residual is sufficiently small, i.e., $||\mathbf{R}(\mathbf{U}^n)||_2 < \epsilon||\mathbf{R}(\mathbf{U}^0)||_2$ where ϵ is some small parameter (typically, $\epsilon \approx 10^{-12}$–$10^{-10}$).

Each step of the Newton iterations requires the solution of a system of linear equations given by Eq. (7) which can be re-expressed as

$$\mathbf{J}\mathbf{x} = \mathbf{b}\,, \tag{9}$$

where $\mathbf{x}=\Delta\mathbf{U}$ and $\mathbf{b}=-\mathbf{R}(\mathbf{U})$ are the solution and right-hand-side vectors, respectively. For most practical flow computations, this system is large, sparse, and non-symmetric and iterative methods can be very effective in its solution. Application of an iterative technique leads to an overall solution algorithm with iterations within iterations: the "inner loop" iterations involving the solution of the linear system and the "outer loop" iterations associated with the solution of the nonlinear problem. An inexact Newton method is adopted here in which the inner iterations are not fully converged at each Newton step. The inner iterations are carried out only until $||\mathbf{R}^n+\mathbf{J}^n\Delta\mathbf{U}^n||_2 \leq \zeta||\mathbf{R}^n||_2$, where ζ is typically in the range 0.1–0.5. As discussed by Dembo *et al.*,[42] an exact solution of the linear system is not necessary for rapid convergence of Newton's method.

D. Parallel GMRES with Schwarz Preconditioning

The class of Krylov subspace iterative methods known as GMRES methods has been developed by Saad and co-workers[15–18] and used extensively in many applications for the solution of large sparse non-symmetric linear equations.[19–24] For many practical applications the Jacobian, $\mathbf{J}$, is however ill-conditioned and preconditioning is required for GMRES to be effective. Although the preconditioner can be applied from either side of $\mathbf{J}$, right preconditioning will be considered here:

$$(\mathbf{J}\mathbf{M}^{-1})(\mathbf{M}\mathbf{x}) = \mathbf{b}\,, \tag{10}$$

where $\mathbf{M}$ is the preconditioning matrix. Furthermore, a convenience of right preconditioning is that the solution residual is unaffected by the preconditioning. Saad[18] indicates that the choice of the side for the preconditioner should not significantly impact GMRES convergence, provided $\mathbf{M}$ is itself not poorly conditioned.

A variety of preconditioning methods are possible. The ideal preconditioner, $\mathbf{M}$, will provide a good approximation to $\mathbf{J}^{-1}$ ($\mathbf{M}^{-1}\approx\mathbf{J}^{-1}$) while being significantly easier to invert than $\mathbf{J}$. Obviously, there is a trade-off between the cost of constructing and applying the preconditioner and the gain in convergence rate of the GMRES algorithm. In the proposed algorithm, a combination of global and local preconditioning techniques is used. In particular, an additive Schwarz global preconditioner with variable overlap is used in conjunction with block-fill incomplete lower-upper (BFILU) local preconditioning. This combination of preconditioning fits well with the block-based AMR described above and is compatible with domain decomposition methods, readily enabling parallel implementation of the overall Newton method. Rather efficient parallel implementations of implicit algorithms via Schwarz preconditioning have been developed by Keyes and co-researchers and successfully applied to the prediction of transonic full potential, low-Mach-number compressible combusting, and three-dimensional inviscid flows.[25–28]

Additive Schwarz preconditioning is similar to a block-Jacobi procedure. The solution on each subdomain or solution block can be updated simultaneously, in a parallel fashion, and shared boundary data on the subdomains is not updated until a full cycle of updates is completed on all domains. The global additive Schwarz preconditioner for N_{blocks} solution blocks can be defined as follows:

$$\mathbf{M}^{-1} = \sum_{k=1}^{N_{\text{blocks}}} \mathbf{B}_k^{\mathrm{T}}\mathbf{M}_k^{-1}\mathbf{B}_k\,, \tag{11}$$

where $\mathbf{B}_k$ is the gather operator or matrix for the k^{th} domain that gathers the solution unknowns for the domain from the global solution vector. In general, domain overlap is permitted. The use of overlapping subdomains can help to offset the loss of overall implicitness of the Newton iterative solver introduced by the block-based Schwarz preconditioning. In Eq. (11), $\mathbf{M}_k^{-1}$ is the local block preconditioner for the subdomain k. Here the local preconditioner is based on ILU factorization[18] of the Jacobian of the first-order (fully

limited) upwind scheme for each sub-domain, $\tilde{\mathbf{J}}_k$. A block-fill ILU(f) or BFILU(f) factorization of $\tilde{\mathbf{J}}_k$ is used where $\mathbf{M}_k$ is given by

$$\mathbf{M}_k = \mathbf{L}_k \mathbf{U}_k \approx \tilde{\mathbf{J}}_k \,, \tag{12}$$

and where $\mathbf{L}_k$ and $\mathbf{U}_k$ are sparse lower and upper triangular matrices. The accuracy of the incomplete LU factorization is determined by the level of fill, f. With higher fill levels, more non-zero entries are retained in the $\mathbf{L}_k$ and $\mathbf{U}_k$ factors above the sparsity pattern of the local Jacobian matrix providing a more accurate representation for $\tilde{\mathbf{J}}_k$; however, this is at the cost of greater computational work and storage. Although the existence and stability of ILU(f) factorizations has only been established for a restricted class of matrices,[43] the approach has been applied to a wide range of systems and McHugh et al.[27] and Gropp *et al.*[28] have shown that ILU factorization can be an effective local preconditioner for parallel NKS algorithms.

A restarted version of the GMRES algorithm, GMRES(m), is used here to solve the additive Schwarz, BFILU(f) preconditioned system of linear equations given in Eq. (10). Although the iterative GMRES algorithm is guaranteed to converge for well-conditioned systems in at most N steps or Krylov subspace search directions, where N is the system size, the storage requirements increase linearly with the number of steps and the computational work increases quadratically. To provide control of storage, Saad and Schultz[16] devised a variant of the GMRES algorithm, restarted GMRES(m), where the GMRES procedure is restarted every m steps. Care must be taken in selecting a value for m to ensure good convergence properties of the GMRES and Newton iterative methods. The application of the Schwarz preconditioning is straightforward as the multi-block quadrilateral mesh has a natural block structure and provides a simple partitioning of the problem into subdomains. The Schwarz preconditioning is in fact quite transparent and achieved once the solution blocks have been established. The main challenge is to perform the necessary inter-block communication to ensure that the global linear problem is solved.

The GMRES algorithm does not explicitly require the evaluation of the global Jacobian matrix, $\mathbf{J} = \partial\mathbf{R}/\partial\mathbf{U}$. It only requires the evaluation of the matrix-vector product $\mathbf{J}\mathbf{M}^{-1}\mathbf{x}$. Numerical differentiation based on Fréchet derivatives provides an approximate expression for this matrix-vector product:[17,21–23,25,27]

$$\mathbf{J}\mathbf{M}^{-1}\mathbf{x} \approx \frac{\mathbf{R}(\mathbf{U} + \varepsilon\mathbf{M}^{-1}\mathbf{x}) - \mathbf{R}(\mathbf{U})}{\varepsilon} \,, \tag{13}$$

where $\mathbf{R}(\mathbf{U} + \varepsilon\mathbf{M}^{-1}\mathbf{x})$ is the residual vector evaluated at some perturbed solution state and ε is a small scalar quantity. Use of this approximation yields a so-called "matrix-free" or "Jacobian-free" inexact Newton method. Although the performance of the matrix-free method is sensitive to the choice of ε, Neilsen *et al.*[21] have found that $\varepsilon = \varepsilon_\circ / ||\mathbf{x}||_2^{1/2}$ seems to work well, with $\varepsilon_\circ \approx 10^{-8}$–$10^{-7}$, and this expression is used here.

The proposed matrix-free inexact Newton method with additive Schwarz/BFILU preconditioning and restarted GMRES linear solver is a variant of the NKS algorithm of Gropp *et al.*[28] From the preceding discussion, it is evident that there are a number of parameters that can affect the numerical performance and convergence of this iterative method, including subdomain overlap, fill level, f, number of GMRES search directions, m, and convergence tolerance for the inner GMRES iterations, ζ. Other techniques such as slope limiter and residual Jacobian "freezing" can be adopted to reduce computational work and improve convergence. In general, it was found that m and ζ should be set to avoid restarts for optimal computational performance. Gropp *et al.* suggest that fill levels $f = 0$, 1, or 2 with zero domain overlap is an effective strategy.[28] The influence of subdomain overlap was experimented with here and it was also found that for fill levels of 0, 1, and 2, overlap did not provide any significant benefits. With higher levels of fill, overlap was found to provide some benefits, but even then the computational savings were quite modest, at least for the applications considered in the present work. In general, fill levels of 3 and 4 with zero domain overlap were found to provide slightly superior performance to the lower fill levels, but for values of f greater than five, the overall performance of the NKS method, in terms of computational time, was found to deteriorate due to the greater costs of forming and using the ILU preconditioner. Slope limiter freezing was also found to be quite helpful in avoiding convergence stall and is used in most cases considered herein.

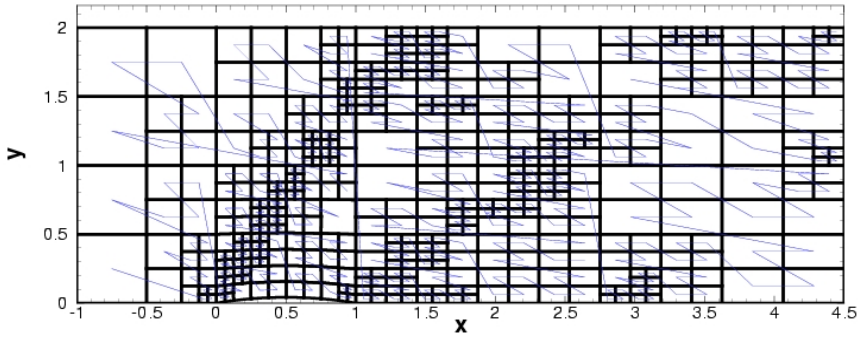


Figure 4. Morton ordering space filling curve used to provide nearest-neighbour ordering of blocks for more efficient load balancing of blocks on multiple processors. The coloured blue line represents the space filling curve passing through each of the solution blocks in the multi-block AMR mesh.

E. Multigrid Startup Algorithm

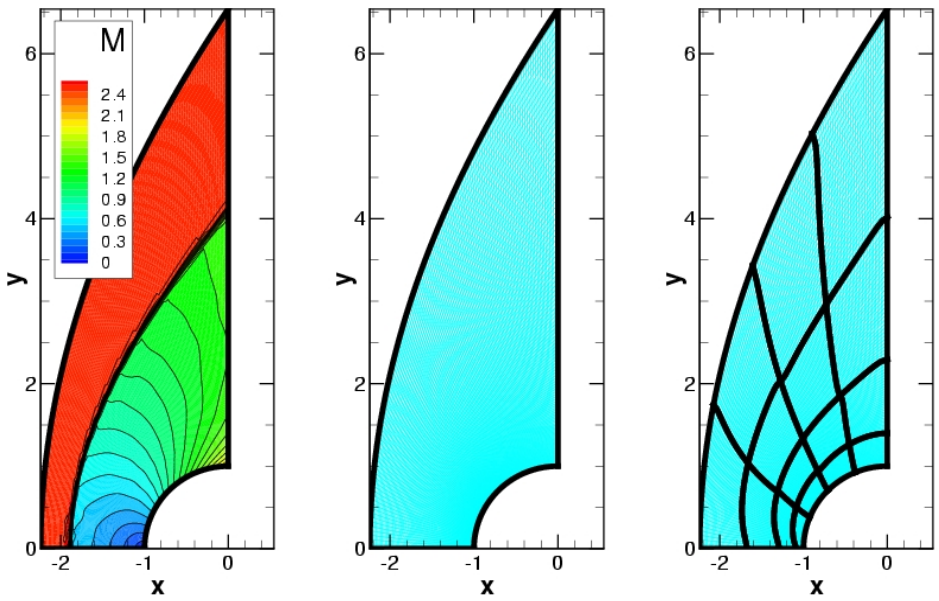
In order to increase the radius of convergence and ensure global convergence of the NKS method for almost any initial data, a good startup startup algorithm is invariably required. Several different startup strategies have been proposed in the literature.[20,21,23] One approach that is often adopted and has proved to be an effective startup procedure is an implicit Euler time-marching method with switched evolution/relaxation (SER) as proposed by van Leer and Mulder.[44] The application of an implicit Euler time-marching method to the semi-discrete form of the governing equations given in Eq. (4) yields

$$\left[-\frac{\mathbf{I}}{\Delta t^n} + \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^n\right] \Delta \mathbf{U}^n = -\mathbf{R}^n \,. \tag{14}$$

where Δt^n is the time step. As $\Delta t^n \to \infty$, Newton's method of Eq. (7) is recovered. In the SER approach, the time step is varied, starting from a finite-value and gradually increasing and becoming very large as the desired steady solution is approached. As the time step approaches infinity Newton convergence is achieved. In the present work, a parallel full multigrid algorithm with both V- and W-cycles has been been developed using the underlying finite-volume discretization procedure coupled with the optimally-smoothing multi-stage time marching schemes developed by van Leer *et al.*[45] as a smoother. See Li[46] for a description of the multigrid solution algorithm. The parallel multigrid method is used as a startup procedure for the NKS algorithm to obtain a good initial estimate for the solution. A small number of full multigrid cycles are used to reduce the solution residual by one- or two-orders in magnitude before initiating the NKS solution procedure. The multigrid strategy seems to be very effective in ensuring the global convergence of the proposed algorithm and is fully compatible with the block-based AMR scheme and parallel implementation advocated herein.

F. Parallel Implementation

By design, the multi-block body-fitted AMR scheme and NKS algorithm are well suited to parallel implementation on distributed-memory multi-processor architectures. A parallel implementation of the block-based AMR scheme has been developed using the C++ programming language and the MPI (message passing interface) library. For homogeneous architectures with multiple processors all of equal speed, the self-similar nature of the solution blocks is exploited and parallel implementation is carried out via domain decomposition where the solution blocks are simply distributed equally among the available processors, with more than one block permitted on each processor. A Morton ordering space filling curve, as shown in Figure 4, is used to provide nearest-neighbour ordering of blocks for more efficient load balancing.[4] A simple stack is used to



(a) (b) (c)

Figure 5. Computed solution of parallel NKS algorithm for supersonic flow past a circular cylinder; Mach number $M_\infty = 2.5$; showing (a) Mach number distribution and (b) single (128×128) and (c) 16 (32×32) block, 16,384 cell, meshes used in computations.

keep track of available (open or unused) processors. For heterogeneous machines, such as a computational grids, a weighted distribution of the blocks can be adopted to preferentially place more blocks on the faster processors and less blocks on the slower processors.

In order to carry out mesh refinement and inter-block communication, a complete copy of the hierarchical quadtree data structure is stored on each processor. This is possible because, unlike cell-based unstructured meshing techniques, the block-based tree data structure is not overly large. The structure need only retain the connectivity between the solution blocks as opposed to a complete map of the cell connectivity required by general unstructured mesh procedures. Inter-processor communication is mainly associated with block interfaces and involves the exchange of ghost-cell solution values, residuals, and conservative flux corrections at every step of GMRES and Newton iterative procedures. Message passing of the this information is performed in an asynchronous fashion with gathered wait states and message consolidation.

IV. Numerical Results

The numerical results are now described for several flow problems. Supersonic flow past a cylinder and transonic and supersonic flow in a channel containing a bump on the lower surface are considered.

A. Supersonic Flow Past a Cylinder

The first test case considered in the evaluation of the proposed parallel NKS algorithm is that of supersonic flow past a two-dimensional circular cylinder with its axis of symmetry perpendicular to the free-stream flow direction such that a stationary bow shock forms about the body. The free-stream Mach number is

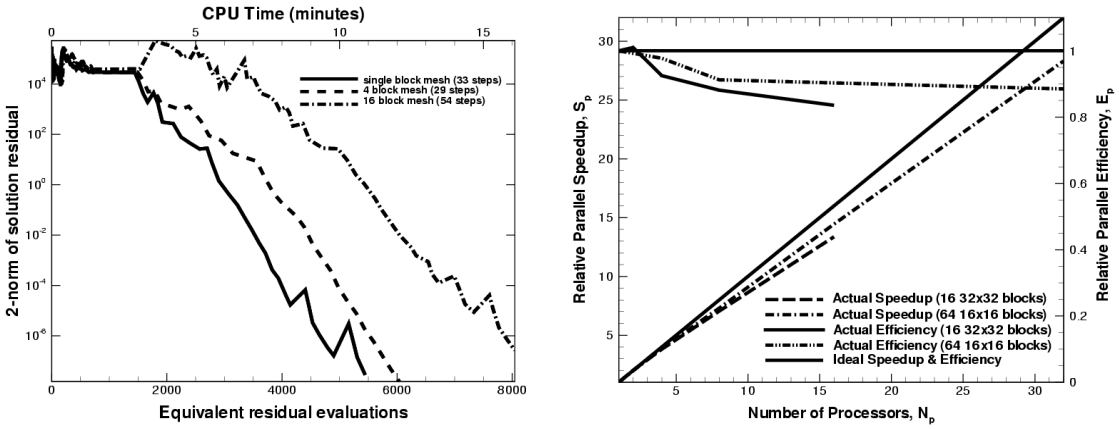


Figure 6. Performance of parallel NKS algorithm for supersonic flow past a circular cylinder; Mach number M=2.5; (a) convergence history showing 2-norm of density residual as a function of the number of equivalent residual evaluations and total processor time and (b) relative parallel speed-up, $S_p = (t_1/t_p)p$, and parallel efficiency, E_p, as a function of the number of processors used in the calculation.

$M_\infty = 2.5$ for the case of interest. Numerical solutions are calculated on a 128×128 mesh consisting of 16,384 computational cells and several different blockings (partitionings) of the computational domain are examined in order to investigate the influence of the Schwarz preconditioning. A single-block grid consisting of one 128×128 solution block is considered, along with 4-, 16-, and 64-block grids composed of 4 64×64, 16 32×32, and 64 16×16 solutions blocks. The single- and 16-block computational mesh are shown in Figures 5(b) and 5(c) for comparison. Mesh refinement is not considered for this problem.

The computed Mach number distribution for the $M_\infty = 2.5$ blunt-body flow is shown in Figure 5(a) and the convergence of the parallel NKS algorithm is given in Figure 6(a), where the 2-norm of density residual is depicted as a function of the number of equivalent residual evaluations and total processor time residual evaluation. The latter are related to one another by the CPU time required to evaluate the residual vector, $\mathbf{R}$, on the finest mesh. Convergence results are shown for the 1-, 4-, and 16-block cases in Figure 6(a). In each of these cases, a 4-level, V-cycle full multigrid procedure, with 100 multigrid cycles on each successively finer mesh, was used to provide a good estimate for the solution before initiating the NKS iterations. For the NKS steps, the values of various solution parameters were $\zeta = 0.2$, $m = 40$, $f = 4$, and no or zero domain overlap was employed in the Schwarz preconditioning. In addition, the values of the slope limiters were "frozen" and held constant after the norm of the solution residual was reduced by four orders in magnitude.

The computed Mach number distributions of Figure 5(a) demonstrate the prediction capabilities of the proposed algorithm for this class of flow problem. The structure and position of the bow shock are well resolved and the subsonic region in the vicinity of the stagnation point on the cylinder is accurately represented. It is also evident from the convergence histories Figure 6(a) that the multigrid startup algorithm is very effective in providing a good initial estimate of the steady solution for the parallel NKS algorithm. Furthermore, the matrix-free inexact Newton method rapidly converges to the steady-state solution. Just 33 Newton steps are required to reduce the solution residual by 12 orders in magnitude on the single block 128×128 mesh. This number drops slightly to 29 for the 4-block mesh and only increases to 54 for the 16-block mesh. It is clear that there is some deterioration of the Newton method performance produced by the increased partitioning used in the additive Schwarz preconditioning. The results for this case indicate that the 16-block partitioning leads to approximately a 40% increase in the overall computational costs of the Newton method as compared to the single partition case. Note that in all cases, the number of GMRES iterations required to reduce the residual for the linear problem by a factor of $\zeta = 0.2$ is typically between 10 and 30. In some cases the number of search directions can be fewer than 5 and, in only a few instances,

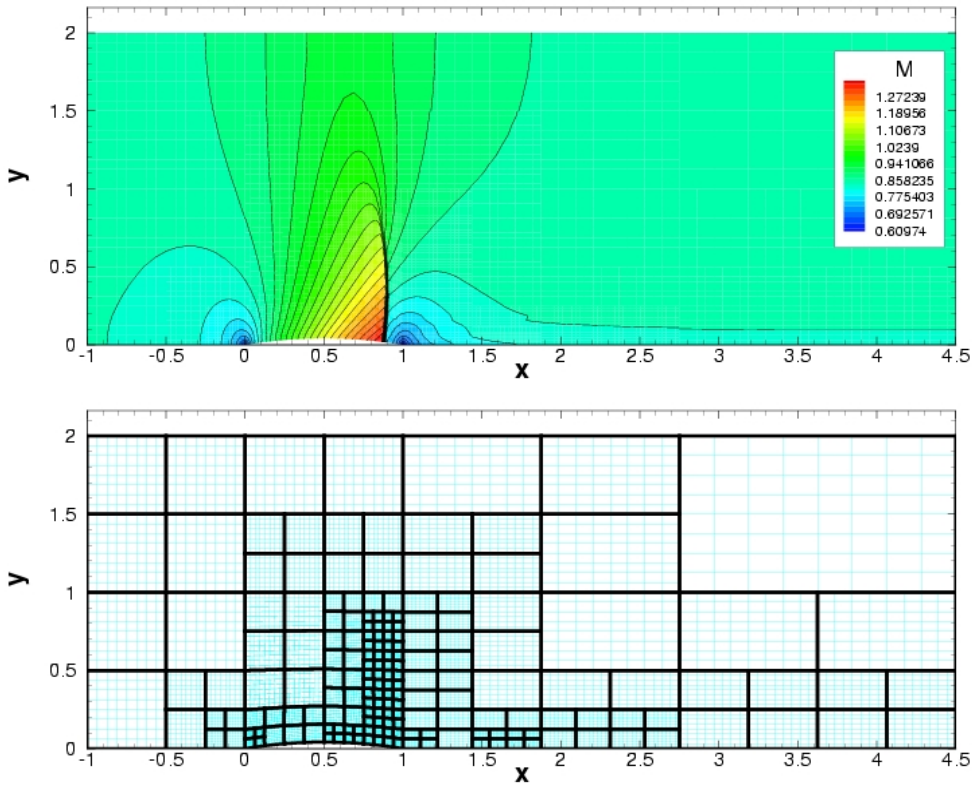


Figure 7. Computed solution of parallel NKS algorithm for transonic channel flow past a bump; Mach number M=0.8; showing Mach number distribution and adapted mesh after 6 levels of refinement; final mesh consists of 182 (8×8) blocks and 11,648 computational cells.

does the iteration count exceed $m=40$ requiring a restart.

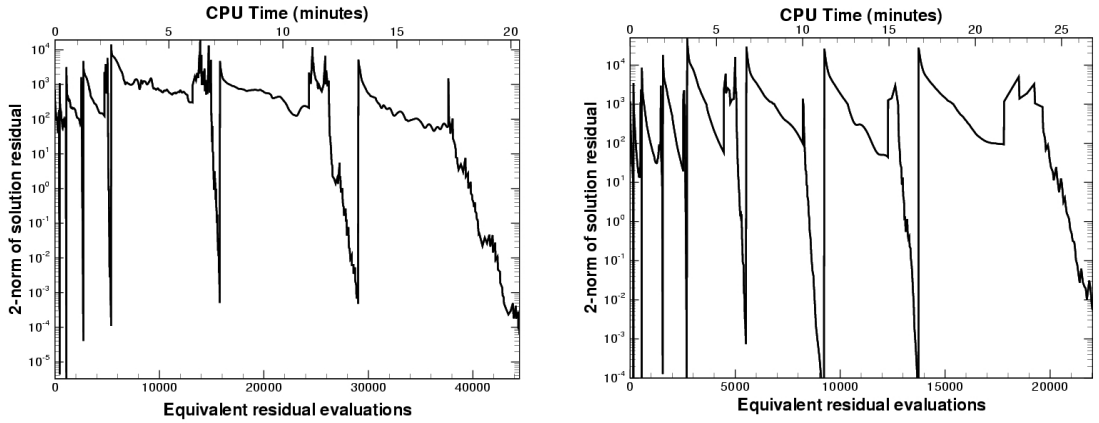
Parallel implementation of the proposed algorithm has been carried out on parallel cluster of 4-way Hewlett-Packard ES40, ES45, and Integrity rx4640 servers with a total of 244 Alpha and Itanium 2 processors. A low-latency Myrinet network and switch is used to interconnect the servers in the cluster. Estimates of the parallel performance and scalability of the proposed method on this parallel architecture for the blunt-body flow problem are shown in Figure 6(b). The figure depicts both the relative parallel speed-up, S_p, given by

$$S_p = \frac{t_1}{t_p} p \,, \tag{15}$$

and the relative parallel efficiency, E_p, given by

$$E_p = \frac{S_p}{p} \,, \tag{16}$$

for a fixed-size problem (fixed total computational work) as a function of the number of processors, p, where t_p is the total processor time required to solve the problem using p processors and t_1 is the processor time required to solve the problem using a single processor. Two separate performance curves are shown and compared to the idealized values of each performance measure for up to 32 processors. The results correspond to two different domain decomposition scenarios for the blunt-body flow problem: the 16- and 64-block partitionings of the computational domain. It is evident from the performance curves of Figure 6(b),



(a) (b)

Figure 8. Convergence histories of parallel NKS algorithm for transonic and supersonic channel flow past a bump; showing (a) 2-norm of density residual as a function of the number of equivalent residual evaluations and total processor time for Mach number M$=0.8$ with six successively refined adapted mesh consisting of from 512 to 11,648 computational cells and (b) 2-norm of density residual as a function of the number of equivalent residual evaluations and total processor time for Mach number M$=1.4$ with seven successively refined adapted mesh consisting of from 512 to 32,000 computational cells.

that while there are indeed some inefficiencies in parallel implementation of the NKS algorithm and parallel performance for larger numbers of processors remains to be fully investigated, the relative parallel speedup is nearly linear in both cases and the parallel efficiency remains above 85%. It is felt that further investigation of the influences of the various solution parameters is required to fully optimize the parallel performance of the algorithm. Nevertheless, these results are certainly very encouraging.

B. Transonic Flow Past a Bump in a Channel

The application of the parallel implicit AMR algorithm to predicting steady inviscid flows through a channel with a bump is now considered. The bump flow problem consists of a rectangular shaped flow domain with a circular-arc-shaped bump on the south boundary. A transonic flow case is first considered where the upstream channel inlet flow Mach number is M$=0.8$.

The initial grid for this problem consisted of eight 8×8 solution blocks with a total of 512 cells. Six levels of mesh refinement were performed and the final mesh, shown in Figure 7, consisted of 182 blocks and 11,648 cells. The predicted Mach number distribution on the final mesh is also given in Figure 7, depicting the steady shock structure which forms at the rear of the bump. The convergence of the combined multigrid startup and parallel NKS algorithm on the seven grids for this case is shown in Figure 8(a). The solution parameters used in this case are similar to those used in the blunt-body flow problem described above. Rapid convergence of the Newton method is obtained on each mesh level. The number of Newton steps required to reduce the residual by eight orders in magnitude varies from 25 to 65, with a maximum of 65 Newton steps required on the finest mesh, and this is for a relatively fine partitioning of the problem where there can be hundreds of subdomains and each subdomain consists only of 8×8 cells.

The computational mesh of Figure 7 illustrates the ability of the body-fitted multi-block AMR procedure to adapt automatically to solution features without significant user input. Mesh refinement is carried out in the vicinity of the rear of the bump, providing accurate resolution of the stationary shock that forms above the surface of the bump. A measure of the efficiency of the block-based AMR scheme for this problem can be defined by a refinement efficiency parameter, η, given by

$$\eta = 1 - N_{\text{cells}}/N_{\text{uniform}} \tag{17}$$

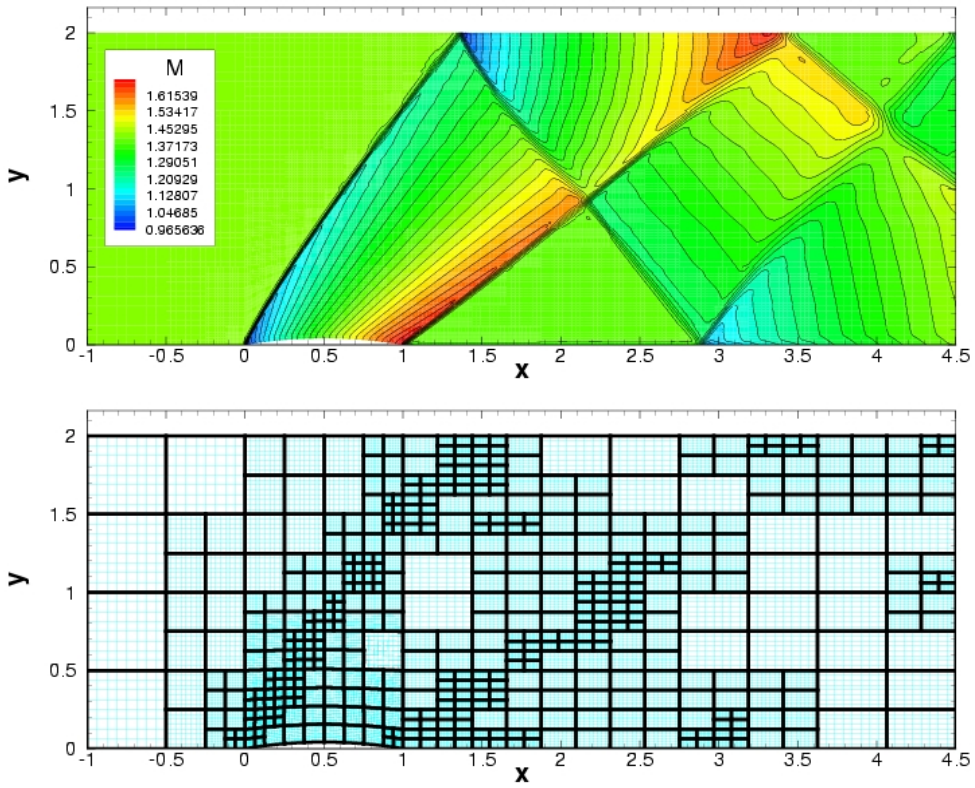


Figure 9. Computed solution of parallel NKS algorithm for supersonic channel flow past a bump; Mach number M=1.4; showing Mach number distribution and adapted mesh after 7 levels of refinement; final mesh consists of 500 (8×8) blocks and 32,000 computational cells.

where N_{cells} is the actual number of cells in the mesh and N_{uniform} is the total number of cells that would have been used on a uniform mesh composed of solution blocks all at the finest level. The efficiency of the AMR scheme is $\eta = 0$ for the initial mesh, where all solution blocks at the same level of refinement, but rapidly improves as the number of refinement levels increases. A refinement efficiency of $\eta = 0.91$ is achieved on the finest mesh, indicating the ability of the block-based AMR approach to deal with flows having disparate spatial scales by reducing the number of computational cells required to solve a problem while maintaining solution resolution in areas of interest.

C. Supersonic Flow Past a Bump in a Channel

Supersonic channel flow is now considered where the upstream channel inlet flow Mach number is M = 1.4. The initial grid again consisted of eight 8×8 solution blocks with 512 cells. Seven levels of mesh refinement were performed and the final mesh, given in Figure 9, consisted of 500 blocks and 32,000 cells. The shocks that form at the leading and trailing edges of the bump and the reflected shock structures that arise at the upper surface of the channel are all depicted in the computed Mach number distribution of Figure 9. Convergence of the parallel NKS method on the eight grids is given in Figure 8(b). Again, rapid convergence of the Newton method is observed for each mesh level, with the number of Newton steps required to reduce the residual by eight orders in magnitude varying from 22 to a maximum of 51. The computational mesh of Figure 9 further illustrates the capabilities of body-fitted multi-block AMR procedure. Accurate resolution of the multiple shocks is achieved with a mesh refinement efficiency of $\eta = 0.76$ on the finest grid.

V. Conclusions

A highly parallelized implicit AMR algorithm has been developed for predicting two-dimensional compressible gaseous flows on body-fitted multi-block mesh. The parallel implicit method allows for anisotropic mesh refinement and appears to be well suited for predicting complex flows with disparate spatial and temporal scales in a reliable and efficient fashion. Future research will focus on the further development of the algorithm for unsteady and three-dimensional flows and for more complex systems of equations. In addition, alternative preconditioning techniques will be explored, including multi-level preconditioning techniques. The possibility of re-using or re-cycling the Krylov subspaces to reduce computational costs as proposed by Gosselet and Rey[47] and Parks *et al.*[48] will also be investigated.

VI. Acknowledgments

This research was supported by a Premier's Research Excellence Award from the Ontario Ministry of Energy, Science, and Technology and by the Natural Sciences and Engineering Research Council of Canada. Funding for the parallel computing facility used to perform the computations described herein was obtained from the Canadian Foundation for Innovation and Ontario Innovation Trust (CFI Project No. 2169). The authors are very grateful to these funding agencies for this support.

References

[1]Berger, M. J., "Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations," *Journal of Computational Physics*, Vol. 53, 1984, pp. 484–512.

[2]De Zeeuw, D. and Powell, K. G., "An Adaptively Refined Cartesian Mesh Solver for the Euler Equations," *Journal of Computational Physics*, Vol. 104, 1993, pp. 56–68.

[3]Aftomis, M. J., Berger, M. J., and Melton, J. E., "Robust and Efficient Cartesian Mesh Generation for Component-Base Geometry," *AIAA Journal*, Vol. 36, No. 6, 1998, pp. 952–960.

[4]Aftomis, M. J., Berger, M. J., and Murman, S. M., "Applications of Space-Filling Curves to Cartesian Methods for CFD," Paper 2004-1232, AIAA, January 2004.

[5]Quirk, J. J. and Hanebutte, U. R., "A Parallel Adaptive Mesh Refinement Algorithm," Report 93-63, ICASE, August 1993.

[6]Berger, M. J. and Saltzman, J. S., "AMR on the CM-2," *Applied Numerical Mathematics*, Vol. 14, 1994, pp. 239–253.

[7]Groth, C. P. T., Zeeuw, D. L. D., Powell, K. G., Gombosi, T. I., and Stout, Q. F., "A Parallel Solution-Adaptive Scheme for Ideal Magnetohydrodynamics," Paper 99-3273, AIAA, June 1999.

[8]Groth, C. P. T., De Zeeuw, D. L., Gombosi, T. I., and Powell, K. G., "Global Three-Dimensional MHD Simulation of a Space Weather Event: CME Formation, Interplanetary Propagation, and and Interaction with the Magnetosphere," *Journal of Geophysical Research*, Vol. 105, No. A11, 2000, pp. 25,053–25,078.

[9]Davis, R. L. and Dannenhoffer, J. F., "Decomposition and Parallelization Strategies for Adaptive Grid-Embedding Techniques," *International Journal of Computational Fluid Dynamics*, Vol. 1, 1993, pp. 79–93.

[10]Sun, M. and Takayama, K., "Conservative Smoothing on an Adaptive Quadrilateral Grid," *Journal of Computational Physics*, Vol. 150, 1999, pp. 143–180.

[11]Hartmann, R. and Houston, P., "Adaptive Discontinuous Galerkin Finite Element Methods for the Compressible Euler Equations," *Journal of Computational Physics*, Vol. 183, 2002, pp. 508–532.

[12]Sachdev, J. S., Groth, C. P. T., and Gottlieb, J. J., "Parallel Solution-Adaptive Scheme for Multi-Phase Core Flows in Rocket Motors," Paper 2003–4106, AIAA, June 2003.

[13]Northrup, S. A. and Groth, C. P. T., "Solution of Laminar Diffusion Flames Using a Parallel Adaptive Mesh Refinement Algorithm," Paper 2005–0547, AIAA, January 2005.

[14]Sachdev, J. S., Groth, C. P. T., and Gottlieb, J. J., "A Parallel Solution-Adaptive Scheme for Predicting Multi-Phase Core Flows in Solid Propellant Rocket Motors," *International Journal of Computational Fluid Dynamics*, Vol. 19, No. 2, 2005, pp. 157–175.

[15]Saad, Y. and Schultz, M. H., "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Equations," *SIAM Journal for Scientific and Statistical Computing*, Vol. 7, No. 3, 1986, pp. 856–869.

[16]Saad, Y., "Krylov Subspace Methods on Supercomputers," *SIAM Journal for Scientific and Statistical Computing*, Vol. 10, No. 6, 1989, pp. 1200–1232.

[17]Brown, P. N. and Saad, Y., "Hybrid Krylov Methods for Nonlinear Systems of Equations," *SIAM Journal for Scientific and Statistical Computing*, Vol. 11, No. 3, 1990, pp. 450–481.

[18]Saad, Y., *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, Boston, 1996.

[19]Wigton, L. B., Yu, N. J., and Young, D. P., "GMRES Acceleration of Computational Fluid Dynamics Codes," Paper 85-1494, AIAA, July 1985.

[20]Venkatakrishnan, V. and Mavriplis, D. J., "Implicit Solvers for Unstructured Meshes," *Journal of Computational Physics*, Vol. 105, 1993, pp. 83–91.

[21]Nielsen, E. J., Anderson, W. K., Walters, R. W., and Keyes, D. E., "Application of Newton-Krylov Methodology to a Three-Dimensional Unstructured Euler Code," Paper 95-1733-CP, AIAA, June 1995.

[22]Barth, T. J. and Linton, S. W., "An Unstructured Mesh Newton Solver for Compressible Fluid Flow and Its Parallel Implementation," Paper 95-0221, AIAA, January 1995.

[23]Pueyo, A. and Zingg, D. W., "An Efficient Newton-GMRES Solver for Aerodynamic Computations," Paper 97-1955, AIAA, June 1997.

[24]Knoll, D. A. and Keyes, D. E., "Jacobian-Free Newton-Krylov Methods: A Survey of Approaches and Applications," *Journal of Computational Physics*, Vol. 193, 2004, pp. 357–397.

[25]Knoll, D. A., McHugh, P. R., and Keyes, D. E., "Newton-Krylov Methods for Low-Mach-Number Compressible Combustion," *AIAA Journal*, Vol. 34, No. 5, 1996, pp. 961–967.

[26]Cai, X.-C., Gropp, W. D., Keyes, D. E., Melvin, R. G., and Young, D. P., "Parallel Newton-Krylov-Schwarz Algorithms for the Transonic Full Potential Equations," *SIAM Journal on Scientific Computing*, Vol. 19, No. 1, 1998, pp. 246–265.

[27]McHugh, P. R., Knoll, D. A., and Keyes, D. E., "Application of Newton-Krylov-Schwarz Algorithm to Low-Mach-Number Compressible Combustion," *AIAA Journal*, Vol. 36, No. 2, 1998, pp. 290–292.

[28]Gropp, W. D., Kaushik, D. K., Keyes, D. E., and Smith, B. F., "High-Performance Parallel Implicit CFD," *Parallel Computing*, Vol. 27, 2001, pp. 337–362.

[29]Barth, T. J., "Recent Developments in High Order K-Exact Reconstruction on Unstructured Meshes," Paper 93-0668, AIAA, January 1993.

[30]Venkatakrishnan, V., "On the Accuracy of Limiters and Convergence to Steady State Solutions," Paper 93-0880, AIAA, January 1993.

[31]Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, 1981, pp. 357–372.

[32]Einfeldt, B., "On Godunov-Type Methods for Gas Dynamics," *SIAM Journal on Numerical Analysis*, Vol. 25, 1988, pp. 294–318.

[33]Linde, T. J., *A Three-Dimensional Adaptive Multifluid MHD Model of the Heliosphere*, Ph.D. thesis, University of Michigan, May 1998.

[34]Linde, T., "A practical, general-purpose, two-state HLL Riemann solver for hyperbolic conservation laws," *International Journal for Numerical Methods in Fluids*, Vol. 40, 2002, pp. 391–402.

[35]Toro, E. F., Spruce, M., and Speares, W., "Restoration of the Contact Surface in the HLL-Riemann solver," *Shock Waves*, Vol. 4, No. 1, 1994, pp. 25–34.

[36]Gottlieb, J. J. and Groth, C. P. T., "Assessment of Riemann Solvers for Unsteady One-Dimensional Inviscid Flows of Perfect Gases," *Journal of Computational Physics*, Vol. 78, 1988, pp. 437–458.

[37]Berger, M. J. and Colella, P., "Local Adaptive Mesh Refinement for Shock Hydrodynamics," *Journal of Computational Physics*, Vol. 82, 1989, pp. 67–84.

[38]Quirk, J. J., *An Adaptive Grid Algorithm for Computational Shock Hydrodynamics*, Ph.D. thesis, Cranfield Institute of Technology, January 1991.

[39]Sorenson, R. L., "A Computer Program to Generate Two-Dimensional Grids About Airfoils and Other Shapes by the Use of Poisson's Equation," Technical Memorandum 81198, NASA, May 1980.

[40]Thompson, J. F., Warsi, Z. U. A., and Mastin, C. W., *Numerical Grid Generation—Foundations and Applications*, North-Holland, New York, 1985.

[41]Powell, K. G., Roe, P. L., and Quirk, J., "Adaptive-Mesh Algorithms for Computational Fluid Dynamics," *Algorithmic Trends in Computational Fluid Dynmaics*, edited by M. Y. Hussaini, A. Kumar, and M. D. Salas, Springer-Verlag, New York, 1993, pp. 303–337.

[42]Dembo, R. S., Eisenstat, S. C., and Steihaug, T., "Inexact Newton Methods," *SIAM Journal on Numerical Analysis*, Vol. 19, No. 2, 1982, pp. 400–408.

[43]Meijerink, J. A. and van der Voorst, H. A., "An Iterative Solution Method for Linear Systems of Which the Coefficeint Matrix is a Symmetric M-Matrix," *Mathematics of Computation*, Vol. 31, 1977, pp. 148–162.

[44]Mulder, W. A. and van Leer, B., "Experiments with Implicit Upwind Methods for the Euler Equations," *Journal of Computational Physics*, Vol. 59, 1985, pp. 232–246.

[45]van Leer, B., Tai, C. H., and Powell, K. G., "Design of Optimally-Smoothing Multi-Stage Schemes for the Euler Equations," Paper 89-1933-CP, AIAA, June 1989.

[46]Li, E. L., *A Parallel Multigrid Method for Predicting Compressible Flow in Turbomachinery*, Master's thesis, University of Toronto, 2004.

[47]Gosselet, P. and Rey, C., "On a Selective Reuse of Krylov Subspaces in Newton-Krylov Approaches for Nonlinear Elasticity," *Proceedings of the Fourteenth International Conference on Domain Decomposition Methods, Cocoyoc, Mexico, January 6–11, 2002*, edited by D. E. Keyes, O. B. Widlund, and R. Yates, National Autonomous University of Mexico, Mexico City, Mexico, 2003.

[48]Parks, M. L., De Sturler, E., Mackey, G., Johnson, D. D., and Maiti, S., "Recycling Krylov Subspaces for Sequences of Linear Systems," Technical Report UIUCDCS-R-2004-2421, University of Illinois at Urbana-Champaign, March 2004.