Nationaal Lucht- en Ruimtevaartlaboratorium
National Aerospace Laboratory NLR

NLR-TP-2001-363

Multitime multigrid convergence acceleration for periodic problems with future applications to rotor simulations

H. van der Ven, O.J. Boelens and B. Oskam

Nationaal Lucht- en Ruimtevaartlaboratorium
National Aerospace Laboratory NLR

NLR-TP-2001-363

Multitime multigrid convergence acceleration for periodic problems with future applications to rotor simulations

H. van der Ven, O.J. Boelens and B. Oskam

This report is based on a presentation held at the Parallel Computational Fluid Dynamics 2001 Conference, Egmond aan Zee, 21-23 May, 2001.

Division:	Fluid Dynamics
Issued:	October 2001
Classification of title:	Unclassified

Summary

The simulation of certain time-dependent flow phenomena can pose a grand challenge to Computational Fluid Dynamics. In this paper, a multitime multigrid convergence acceleration algorithm is introduced which significantly reduces the turnaround time to reach time-periodic solutions. The application of this convergence acceleration algorithm will lead to time-efficient simulations of helicopter rotors in forward flight. A comparison between multitime multigrid acceleration and the classical serial time multigrid acceleration shows that an order of magnitude reduction in turnaround time can be achieved at the expense of an order of magnitude increase in memory use.

Contents

1 Table

3 Figures

(17 pages in total)

1 Introduction

The simulation of certain time-dependent flow phenomena can pose a grand challenge to Computational Fluid Dynamics (CFD). Turnaround times of time-accurate simulations can be large if the characteristic bandwidth of the time dependent periodic flow solution is large. Bandwidth is defined as the ratio of the maximum and minimum frequency of the Fourier spectrum of the time periodic solution. In this paper, a new convergence acceleration algorithm is introduced which significantly reduces the turnaround time for time-periodic solutions with large bandwidth. The new algorithm will be applied to a space-time discontinuous Galerkin (DG) method (Ref. 10).

1.1 Applications

Main application area for the DG method is the simulation of the flow around helicopter rotors, in both hover and forward flight conditions. Rotor flows are characterised by complicated aerodynamic phenomena, such as blade-vortex interaction and compressibility effects at the advancing blade, leading to high noise levels. Present day CFD technology is sufficiently mature to accurately resolve these aerodynamic phenomena (for details, see Ref. 2), but at a prohibitively high computational cost. For a typical helicopter rotor in steady state forward flight the mimimum frequency is the blade passing frequency, and the maximum frequency is dominated by the short duration Blade-Vortex Interaction events, given rise to a bandwidth of the order of one hundred.

1.2 Parallel algorithm development

For the development of parallel algorithms the ultimate goal is reduction of turnaround time.In a first approximation, turnaround time is computed as work divided by speed, where work is the required number of floating point operations per simulation, and speed is the number of floating point operations per second achieved by the algorithm on a certain architecture. So one can either increase the speed, or decrease the required flop count (increase the algorithm efficiency), in order to decrease the turnaround time per simulation.

It is important to recall that optimising only speed, or work, may lead to suboptimal algorithms. Maximising speed may lead to foolish algorithms, which for instance keep recomputing storable data to increase the flop rate, whereas the turnaround time stays the same. On the other hand, algorithms minimising work may not benefit from the peak speed of existing hardware architectures. If a ‘minimum work’ algorithm is neither vectorisable nor parallelisable, nobody will even consider it. More realistically, dynamic grid algorithms usually are more efficient in terms of flop count since the same solution can be obtained with less grid cells, but the dynamically evolving grids imply load balancing on parallel machines, which is poorly scalable. Hence, parallel algorithm

development should be motivated by the miminisation of the *quotient* work over speed, resulting in a minimum turnaround time.

1.3 Outline

The outline of the paper is as follows. In Section 2 rotor flow simulations are briefly presented. In Section 3 the new convergence acceleration algorithm is presented and results are given in Section 4. Conclusions with respect to forward flight simulations with a minimum turnaround time and parallel algorithm development are drawn in Section 5.

2 Helicopter rotor in forward flight

In recent years at NLR the flow solver based on a discontinuous Galerkin finite element discretisation of the unsteady compressible Euler equations, which was originally designed for fixed wing applications (Ref. 9, 10), has been modified to enable the simulation of helicopter rotor flows (Ref. 1).

The vortex capturing capability through grid adaptation has successfully been used in the simulation of the Caradonna-Tung helicopter rotor in hover, and the simulation of the Operational Loads Survey (OLS) helicopter rotor in forward flight.

Before turning to the forward flight simulation, which is the relevant measure used to assess the minimum turnaround time of a CFD code, the steady simulation of the Caradonna-Tung rotor in hover is discussed, since the efficiency of the grid adaptation algorithm for this simulation is exemplary of what we want to achieve for time-accurate forward flight simulations.

A rotor in hover is considered as a steady state problem in the rotating reference frame. A simulation of the Caradonna-Tung rotor (Ref. 4) using a collective pitch of twelve degrees and a tip Mach number of 0.61 has been performed (for details see Ref. 1). The simulation was performed on both a fine grid and a locally refined coarse grid. The locally refined mesh is shown in Figure 1, where the vortex locations can be clearly seen in the grid structure. Both grids demonstrated the same vortex persistence, with the locally refined mesh containing only 15% of the number of grid cells of the fine mesh. Hence, local grid refinement displays an *algorithmic* speedup of six.

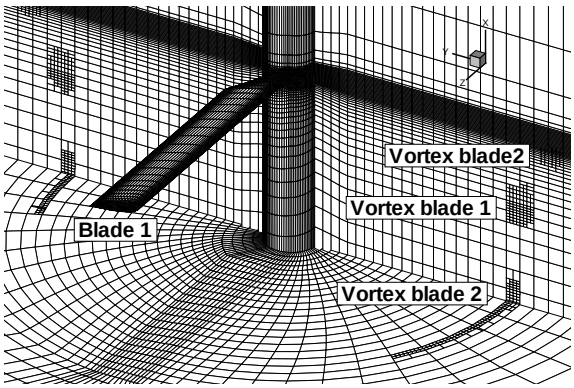


Fig. 1 Local grid refinement based on the vortex sensor applied to the Caradonna-Tung rotor in hover. Shown are the vertical periodic plane and a horizontal cross-section below the rotor blade.

A similar simulation has been performed for the OLS rotor in forward flight, with the difference that this simulation is time-dependent in the inertial frame, and that the vortex position in the grid changes over time. The OLS rotor has been simulated using a tip Mach number of 0.664, an advance ratio of 0.164, and a thrust coefficient of 0.054. The blade motion schedule was obtained by trimming the rotor as a whole using the CFD code, and not a lifting line vortex code. Good agreement with experiments was obtained (Ref. 2).

On the downside, the complete simulation using 288 time steps per period for only three periods on a final refined mesh of 1.2 million grid points took 60 hours (20 hours per period) on the eight processor NEC SX-5/8B, at a congregate speed of 24 Gflop/s (about 40% peak) and using 13 GB of memory. For an analysis tool such a computing time is too large, and inhibits its use in helicopter performance analysis tool sets.

3 Multigrid acceleration

3.1 Background

Multigrid acceleration (Ref. 3) combines classical iterative techniques, such as point relaxation or local time stepping, with coarse level corrections to yield a method superior to the iterative techniques alone. In the present paper two types of multigrid acceleration are considered:

- STMG, serial time multigrid, where the coarse level corrections only pertain to the spatial operator,
- MTMG, multitime multigrid, where the coarse level corrections pertain to both space and time discretisations.

For periodic time-dependent problems STMG acceleration yields an asymptotic convergence rate of $1 - O(\Delta t/T)$, with Δt the time step and T the period, which is the same rate as classical iterative techniques for steady-state problems. MTMG acceleration restores the superior convergence acceleration, where the asymptotic convergence rate is bounded away from one, independent of space and time discretisation. The above will be illustrated in Section 4.

3.2 A multitime multigrid algorithm

Following the ideas of Horton, Vandewalle, and Worley for parabolic equations (Ref. 5, 6, 12) we propose to solve the problem for all time steps simultaneously. Periodic problems, such as the rotor in forward flight, can be considered as steady state problems in the space-time domain, so this would seem like a feasible idea.

Horton and VandeWalle (Ref. 6) considered parabolic PDE's, and introduced a space-time multigrid method in which all time-levels are treated simultaneously. They showed that for parabolic equations it is not possible to treat the time as just another space dimension, and they had to revert to semi-coarsening techniques in order to maintain multigrid convergence. Their restriction and prolongation operators in time take the direction of time into account. As smoother they applied a coloured pointwise Gauss-Seidel relaxation.

These ideas are extended to hyperbolic PDE's in a straightforward way. The equations to be solved remain the same, but all time steps are solved simultaneously. The system can be solved in parallel. The solution strategy of the system is basically the same as for a system of equations derived using a spatial discretisation. A pseudo-time is introduced and the solution is marched to a steady state using standard acceleration techniques such as local time stepping, grid sequencing and multigrid. One important difference is that the multi-level techniques will be applied to the space-time grid, and not be restricted to the space grid only. Hence the method is called a *multitime multigrid*

(MTMG) acceleration algorithm.

In contrast with Horton et al. we treat the time as just another space dimension. The restriction and prolongation operators in time are identical to the space operators. A five stage Runge-Kutta scheme is used as a smoother. No semi-coarsening techniques have been applied in the multigrid algorithm.

Let $\mathcal{L}_h$ be the DG discretisation operator for a given time slab, U_h^n the solution in the time slab $[t^n, t^{n+1}]$, which satisfies the equations

$$\mathcal{L}_h(U_h^n, U_h^{n-1}) = 0,$$

where U_h^{n-1} is the solution in the previous time slab. For a periodic problem the equations are

$$\begin{cases} \mathcal{L}_h(U_h^i, U_h^{i-1}) = 0, & 1 \leq i \leq N, \\ U_h^0 = U_h^N, \end{cases}$$

if the period is divided into N time slabs.

In the STMG algorithm the equations are solved in pseudo-time τ as

$$\frac{\partial \tilde{U}_h^n}{\partial \tau} + \mathcal{L}_h(\tilde{U}_h^n, U_h^{n-1}) = 0, \qquad n = 1, 2, \ldots, \infty$$

where U_h^0 is the inital solution, upon convergence we set $U_h^n = \tilde{U}_h^n$, and proceed to the next time step. In the MTMG approach the equations are solved as

$$\frac{\partial \tilde{U}_h^i}{\partial \tau} + \mathcal{L}_h(\tilde{U}_h^i, \tilde{U}_h^{i-1}) = 0, \qquad 1 \leq i \leq N,$$

simultaneously, with the periodic boundary condition $\tilde{U}_h^0 = \tilde{U}_h^N$.

The L_2-residual ϵ_{per} of a solution obtained with the MTMG algorithm is defined as

$$\epsilon_{\text{per}}^2 = \frac{1}{N} \sum_{i=1}^{N} \|\mathcal{L}_h(\tilde{U}_h^i, \tilde{U}_h^{i-1})\|_2^2, \tag{1}$$

where $\|\cdot\|_2$ is the L_2-norm in the time slab. In the STMG approach the single time step residual $\epsilon_s^{(n)}$ is measured:

$$\epsilon_s^{(n)} = \|\mathcal{L}_h(\tilde{U}_h^n, U_h^{n-1})\|_2, \qquad (n \geq 1) \tag{2}$$

but this residual does not measure the convergence to a periodic solution. Given a series $\tilde{U}_h^{kN+1}$, $\ldots$, $\tilde{U}_h^{(k+1)N}$ of solutions in the k-th period, define the solution vectors $\tilde{V}_h^i = \tilde{U}_h^{kN+i}$, $1 \leq i \leq N$. Considering $\tilde{V}_h$ as a periodic solution, we compute ϵ_{per} as follows:

$$\begin{aligned}
\epsilon_{\text{per}}^2 =& \frac{1}{N}\left(\|\mathcal{L}_h(\tilde{V}_h^1, \tilde{V}_h^N)\|_2^2 + \sum_{i=2}^{N} \|\mathcal{L}_h(\tilde{V}_h^i, \tilde{V}_h^{i-1})\|_2^2\right) \\
=& \frac{1}{N}\left(\|\mathcal{L}_h(\tilde{U}_h^{kN+1}, \tilde{U}_h^{(k+1)N})\|_2^2 + \sum_{i=2}^{N} \|\mathcal{L}_h(\tilde{U}_h^{kN+i}, \tilde{U}_h^{kN+i-1})\|_2^2\right) \\
=& \frac{1}{N}\left(\|\mathcal{L}_h(\tilde{U}_h^{kN+1}, \tilde{U}_h^{(k+1)N})\|_2^2 + \sum_{i=2}^{N} (\epsilon_s^{(kN+i)})^2\right). \qquad (3)
\end{aligned}$$

4 Results

The MTMG method has been applied to two-dimensional transonic flow over an harmonically oscillating NACA0012 foil. The freestream Mach number is 0.8, the angle of attack oscillates between -0.5^o and 4.5^o with a non-dimensional frequency of 0.314. The space-time mesh consists of $256 \times 128 \times 20$ elements, where the last dimension is the time dimension. In Figure 2 the convergence of the MTMG method is compared with that of the conventional STMG method. For the STMG method at each time step the implicit system is solved in pseudo-time, and six time steps are shown. At each time step n the residual $\epsilon_s^{(n)}$ is converged to $5 \cdot 10^{-7}$. For the MTMG method full multigrid has been applied, with 150 iterations on the coarsest mesh with $64 \times 32 \times 5$ grid cells, 200 multigrid cycles on the next finer level, and 500 multigrid cycles on the fine mesh. For both methods V cycles with one prerelaxation and one postrelaxation have been used. The convergence rate of the MTMG method is comparable to the convergence rate of the multigrid algorithm for the space DG discretisation operator for steady state problems.

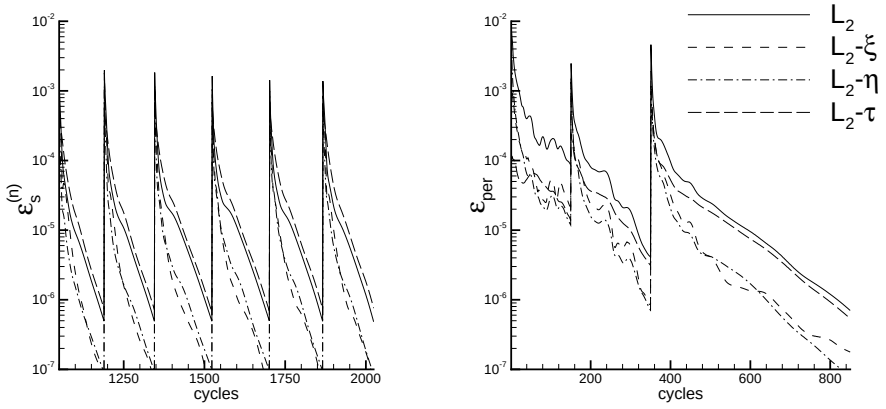
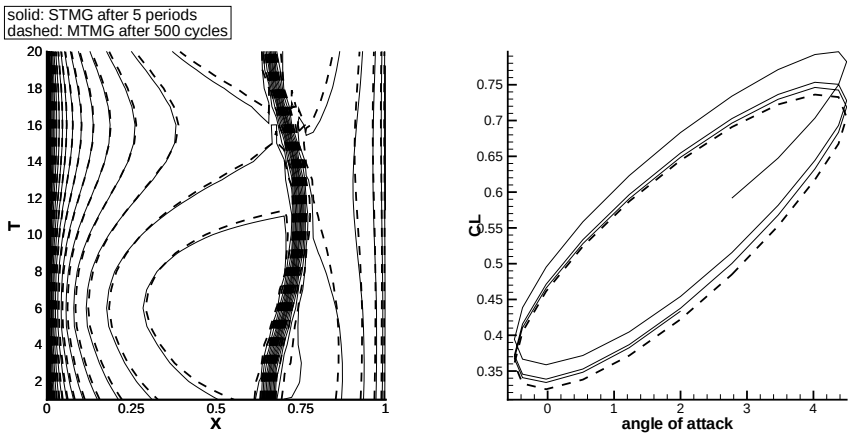


Fig. 2 Convergence history of the residual $\epsilon_s^{(n)}$ of six time steps of the STMG method (left) and the complete convergence history of the residual ϵ_{per} of the MTMG method with full multigrid (right).

The time-dependent pressure distribution on the upper side of the airfoil is shown in Figure 3(a). The motion of the shock is clearly visible. Moreover, the comparison of MTMG with STMG is good. In Figure 3(b) the polar plots of the lift coefficient C_L are shown for both methods: agreement is good, but not excellent, considering the fact that both methods solve the same set of discrete equations. Upon convergence both STMG and MTMG schemes should result in solutions that are equal up to machine accuracy.



(a) Pressure distribution (b) Polar plot of C_L

Fig. 3 Comparison of itwo final iterates, one reached after five periods with STMG, and one after 500 cycles of MTMG

In order to compare the flow solutions of the different methods, we compute the periodic residual for the solution obtained with the STMG method over the first five periods. Using (3) we have

$$\epsilon^2_{\text{per}} = \frac{1}{N}\left(\|\mathcal{L}_h(\tilde{U}_h^{kN+1}, \tilde{U}_h^{(k+1)N})\|_2^2 + (N-1)(5\cdot 10^{-7})^2 \right). \tag{4}$$

In Table 1 this residual is shown for five consecutive periods of the STMG iterates. Clearly, convergence in L_2-norm to a time periodic solution is slow, and the residual $\|\mathcal{L}_h(\tilde{U}_h^{kN+1}, \tilde{U}_h^{(k+1)N})\|_2$ dominates the time step residual $\epsilon_s^{(n)}$. This may cause the differences in the flow results. Note that based on the aerodynamic coefficients in Figure 3 one would conclude convergence in about three periods.

period	ϵ_{per}
1	$0.89 \cdot 10^{-3}$
2	$0.61 \cdot 10^{-3}$
3	$0.35 \cdot 10^{-3}$
4	$0.24 \cdot 10^{-3}$
5	$0.20 \cdot 10^{-3}$

Table 1 Residual ϵ_{per} defined in (4) of the STMG simulation.

Since the discretised equations are the same for both the STMG and the MTMG acceleration algorithms, the number of floating point operations per grid cell per time step per iteration are

equal. There is only negligible difference in the computational cost of the multigrid algorithm, since the coarse grids in the MTMG algorithm are smaller than in the STMG algorithm, since the MTMG grid levels are also coarsened in the time direction. Hence, 150 multigrid cycles for MTMG require the same amount of floating point operations as 150 multigrid cycles per time step for a complete period in the STMG algorithm.

Based on the L_2-norm, the MTMG algorithm would require only 25 fine grid cycles to reach the same residual level as five periods of the STMG algorithm. Since the average number of fine grid cycles per time step of the STMG algorithm is 150, this would imply that MTMG is $(150 \times 5)/25 = 30$ times faster than the STMG algorithm. The speedup is this large since the STMG acceleration performs poorly in terms of reaching the periodic steady state. Since an extensive study of the convergence of time-periodic problems is beyond the scope of the present paper, we will not go into further details. By engineering standards one would require a decrease in the residual of three to four orders in magnitude, depending on the spatial and temporal resolution. To satisfy this standard, the MTMG method requires 250 cycles, while the STMG method would require at least 50 periods to reach the same level of periodicity, again resulting in a speedup of thirty.

The qualitatively greater efficiency of the MTMG method can partly be explained by the fact that it presupposes the existence of a periodic solution, and partly by the fact that the multigrid algorithm is applied to the space-time system, and not only to the space system. Moreover, the full multigrid algorithm provides better initial solutions for the implicit system, whereas the time serial algorithm uses the solution of the previous time step as the initial solution.

5 Conclusions

5.1 Time-periodic simulations

The standard way of obtaining a periodic solution by time integration is a slow process. For an oscillating transonic airfoil, there is hardly any convergence in L_2-norm over five periods, indicating an asymptotic convergence rate of $1 - O(\Delta t/T)$.

Considering the poor convergence of the STMG acceleration, it is difficult to make a definitive comparison between the convergence rate of the MTMG acceleration with the performance of the STMG acceleration.

Considering the computational complexity, MTMG has the following properties:

- the number of periods required to resolve the transient is reduced to one, reducing the work to be done per simulation with a factor in the order of the number of time steps per period,
- the algorithm has increased scalability since the grid size is increased by a factor equal to the number of time steps,
- the memory use increases with a factor proportional to the number of time steps.

The increased scalability and the increase in memory use make the method ideally suited for MPP machines. Especially for time periodic applications with large bandwidth, and for which a large number of time steps is required. The simulation of the rotor in forward flight is such an example, requiring 288 time steps per period.

5.2 Helicopter rotor in forward flight

If we would apply the MTMG algorithm to the simulation of the flow field of a rotor in forward flight, we estimate the following performance increase:

- MTMG versus STMG for k periods yields a speedup of at least k,
- since the time-periodic flow is now treated as a steady state problem in space-time, we can apply local grid refinement to the space-time grid, where the grid is only refined *where* and *when* a vortex is present. A similar reduction in grid size as for the rotor in hover can be expected, yielding a speedup of 6,
- an MPP machine with 1000 processors of 1 Gflop/s each (at a sustained performance of 10% peak speed), would be four times faster than the NEC SX-5/8B (at a sustained performance of 40% peak speed). Since the MTMG algorithm is a static algorithm, it is easily scalable even beyond a 1000 processors, so a speedup of at least four is feasible.

Combining these three improvements, the turnaround time of the simulation of a rotor in forward flight is decreased by a factor $24k$: $20k$ hours for k periods are reduced to less than an hour to

obtain a periodic solution using MTMG. Considering the slow convergence to a periodic solution of the STMG method, one should even doubt that seven periods are sufficient to obtain a periodic solution for the rotor in steady state forward flight, further increasing the speedup of the MTMG method.

Based on the memory requirements of the discontinous Galerkin method (Ref. 11), the unstructured Space-Time DG method requires 241 words per space-time element (the DG method has 25 degrees of freedom per element). Based on the experience with the simulation of the rotor in hover, it is expected that a spatial grid of about 200,000 cells in each time slab is sufficient to accurately capture the tip vortices in that time slab. To obtain such a mesh, the mesh needs to be locally refined in each time slab, to accomodate the movement of the vortices. Hence, the grid adaptation is time-dependent, in the sense that the grid structure is different in each time slab.

In the simulation of the rotor in forward flight 288 time steps per period were performed. Combined with the spatial resolution of 200,000 cells, the space-time grid contains 57.6 million cells, requiring 111 GB memory. This large amount of memory can be reduced if we also locally refine the space-time grid in the time direction. Based on the experience with the simulation of the rotor in hover, local grid refinement results in a reduction in mesh size of approximately a factor two in each dimension. Hence, a simulation of a rotor in forward flight using a locally refined mesh in both space and time would require 55 GB of memory.

5.3 Parallel algorithm development

The MTMG algorithm has shown an algorithmic speedup by a factor of the order of the number of time steps per period with respect to STMG for a two dimensional, time periodic simulation. In the context of parallel computing, however, it is more important that a dynamic algorithm is turned into a static algorithm. *All* grid manipulations are performed in a preprocessing phase, and not at each time step during the simulation. Grid deformation to accomodate the body motion is performed during the grid generation. Local grid refinement has to be performed only two or three times during the simulation, which is the standard procedure for grid refinement for steady state problems.

As an explicit, static method, the MTMG method is easily scalable beyond 1000 processor MPP machines, as has been demonstrated in the American ASCI project (Ref. 7, 8).

Hence, a combination of an increase in algorithm efficiency and algorithm speed is projected to lead to forward flight simulations with a turnaround time of less than an hour on an MPP machine

with 1000 processors of 1 Gflop/s each. Since these kind of machines typically have 1 GB memory per processor, the increased memory requirements of the MTMG acceleration algorithm are not critical to the application.

References

1. O.J. Boelens, H. van der Ven, B. Oskam and A.A. Hassan, *Accurate and efficient vortex-capturing for a helicopter rotor in hover,* in the proceedings of the 26th European Rotorcraft Forum, The Hague, 2000.
2. O.J. Boelens, H. van der Ven, B. Oskam and A.A. Hassan, *The boundary conforming discontinuous Galerkin finite element approach for rotorcraft simulations,* submitted to Journal of Aircraft, 2001.
3. A. Brandt, *Multi-Level adaptive solutions to boundary value problems,* Math. of Comp. **31**, 333-390, 1977.
4. F.X. Caradonna and C. Tung, *Experimental and analytical studies of a model helicopter rotor in hover*, NASA Technical Memorandum 81232, 1981.
5. G. Horton, S. Vandewalle and P. Worley, *An algorithm with polylog parallel complexity for solving parabolic partial differential equations,* SIAM J. Sci. Comput., **16(3)**, 531-541, 1995.
6. G. Horton and S. Vandewalle, *A space-time multigrid method for parabolic PDEs,* SIAM J. Sci. Comput., **16 (4)**, 848-864, 1995.
7. D.E. Keyes, D.K. Kaushik, and B.F. Smith, *Prospects for CFD on Petaflops Systems,* NASA/CR-97-206279, 1997.
8. D.J. Mavripilis, *Large-scale parallel viscous flow computations using an unstructured multigrid algorithm,* NASA/CR-1999-209724, 1999.
9. J.J.W. van der Vegt and H. van der Ven, *Discontinuous Galerkin finite element method with anisotropic local grid refinement for inviscid compressible flows,* J. Comp. Physics, **141**, 46-77, 1998.
10. J.J.W. van der Vegt and H. van der Ven, *Space-Time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows. Part I: General formulation.* Submitted to J. Comp. Physics, 2001.
11. H. van der Ven and J.J.W. van der Vegt, *Accuracy, resolution, and computational complexity of a discontinuous Galerkin finite element method,* in: B.Cockburn and G. Karniadakis and C.-W. Shu (eds.), Discontinuous Galerkin Methods: Theory, Computation and Applications, Lecture Notes in Computational Science and Engineering, **11**, Springer Verlag, Berlin, 2000.
12. P.H. Worley, *Parallelizing across time when solving time-dependent partial differential equations,* in Proc. 5th SIAM Conference on Parallel Processing for Scientific Computing, Eds. J. Dongarra, K. Kennedy, P.Messina, D. Sorensen, and R. Voigt, SIAM, 1992.