DOCUMENT CONTROL SHEET

	ORIGINATOR'S REF.		SECURITY CLASS.
	NLR-TP-99304		Unclassified

ORGINATOR
National Aerospace Laboratory NLR, Amsterdam, The Netherlands

TITLE
Accuracy, resolution, and computational complexity of a discontinuous Galerkin finite element method

PRESENTED AT:
the First International Symposium on Discontinuous Galerkin Methods, Newport, Rhode Islands, U.S.A., 24-26 May 1999.

AUTHORS	DATE	pp	ref
H. van der Ven and J.J.W. van der Vegt* *University of Twente	July 1999	13	8

DESCRIPTORS
Galerkin method
Computational fluid dynamics
Finite element method
Numerical analysis

ABSTRACT
An analysis of the balance between the computational complexity, accuracy, and resolution requirements of a discontinuous Galerkin finite element method for the solution of the compressible Euler equations of gas dynamics is presented. The discontinuous Galerkin finite element method uses a very local discretization, which remains second order accurate on highly non-uniform meshes, but at the cost of an increase in computational complexity and memory use. The question of the balance between computational complexity and accuracy is addressed by studying the evolution of vortices in the wake of a wing. It is demonstrated that the discontinuous Galerkin finite element method on locally refined meshes can result in a significant reduction in computational cost.

Nationaal Lucht- en Ruimtevaartlaboratorium
National Aerospace Laboratory NLR

NLR-TP-99304

Accuracy, resolution, and computational complexity of a discontinuous Galerkin finite element method

H. van der Ven and J.J.W. van der Vegt*

* *University of Twente*

This report is based on a presentation held at the First International Symposium on Discontinuous Galerkin Methods, Newport, Rhode Island, U.S.A., 24-26 May 1999.

Division:	Fluid Dynamics
Issued:	July 1999
Classification of title:	unclassified

Contents

2 Tables
3 Figures

(13 pages in total)

Summary

An analysis of the balance between the computational complexity, accuracy, and resolution requirements of a discontinuous Galerkin finite element method for the solution of the compressible Euler equations of gas dynamics is presented. The discontinuous Galerkin finite element method uses a very local discretization, which remains second order accurate on highly non-uniform meshes, but at the cost of an increase in computational complexity and memory use. The question of the balance between computational complexity and accuracy is addressed by studying the evolution of vortices in the wake of a wing. It is demonstrated that the discontinuous Galerkin finite element method on locally refined meshes can result in a significant reduction in computational cost.

1 Introduction

The accurate calculation of small scale flow structures presents a great challenge to computational fluid dynamics. Wake vortices, shocks, and the viscous sublayer in wall-bounded flows require a resolution which is orders of magnitude finer than in other regions of the flow. Efficient simulation of such structures is only feasible on highly non-uniform meshes, which are refined in the regions of interest. Accurate simulation of the flow structures on locally refined meshes is possible using Discontinuous Galerkin (DG) methods.

Discontinuous Galerkin finite element methods result in a very local discretization, which combines well with h-refinement because it maintains accuracy on non-smooth grids. The discontinuous Galerkin finite element method is, however, considerably more expensive, both in terms of computational complexity and memory usage, in comparison with the more commonly used finite volume methods. The key question to be addressed in this paper is whether for specific fluid dynamics problems, with vastly different length scales in two or more directions, the computational complexity of the DG method is more than compensated by its accuracy.

The balance between accuracy, resolution, and computational complexity of the DG finite element method is investigated by studying its efficiency in capturing the vortices in the wake of a wing. Numerical dissipation and insufficient grid resolution cause serious problems in capturing vortical structures, and result in a smearing and decay of the vortical structures at some distance behind the wing. For many applications it is very important to be able to trace these vortical structures over a large distance downstream.

The outline of the paper is as follows. After a short description of the algorithm, the computational complexity of the method is analyzed. Subsequently its accuracy on highly non-uniform meshes is assessed for vortical flow. Finally, the balance between computational complexity and accuracy will be addressed.

2 Numerical Method

The numerical method used in the present investigation combines a discontinuous Galerkin discretization for the spatial discretization with a TVD-Runge Kutta time integration method and multigrid acceleration. This technique has received considerable theoretical interest during the last decade. Especially the work of Cockburn, Shu, et al. (Ref. 1, 2), significantly contributed to its theoretical development. In a series of papers van der Vegt and van der Ven (Ref. 4, 5, 6) further developed the DG finite element method into a second order accurate numerical technique for the solution of the three-dimensional Euler equations of compressible gas dynamics on highly non-uniform hexahedral meshes. This method is used in the present investigation.

The most computationally intensive part of the method are the element face flux integrals. The straightforward computation of the face flux integrals requires four point Gauss quadrature rules for second order accuracy. Van der Vegt et al. (Ref. 6) proposed an approximation to the flux integrals of the form

$$\int_S F(U_h) \cdot n\phi_m dx \approx F(\overline{U}_h^S) \cdot \int_S n\phi_m dx,$$

where S is a face, U_h is the state vector of the Euler equations, F is the flux function, n is the face normal, ϕ_m ($0 \leq m \leq 3$) is the m-th basis function in the cell bounding S, and $\overline{U}_h^S$ is the face average. The volume fluxes are approximated likewise. Van der Vegt et al. (Ref. 6) proved that second order accuracy is retained when the geometric terms are computed exactly. These approximations result in a number of flux calculations which is approximately equal to finite volume methods, and in a reduced computational complexity.

The algorithm is efficiently implemented in the program HEXADAP using a face based data structure, which allows full vectorization and parallelization of the code. The parallel performance of the code is further improved with a dynamic domain decomposition technique, which automatically redistributes the elements over the processors after grid adaptation (Ref. 7, 8). The computational efficiency is further improved by local time stepping (with CFL=0.7) and a multigrid convergence acceleration algorithm, which uses a first order accurate scheme on the coarser grid levels.

The DG finite element method results in an accurate discretization, but with increased memory use and a significantly larger computational complexity than finite volume methods. The DG method also solves equations for the three higher moments for all five variables of the Euler equations and stores these variables. This results in 20 degrees of freedom per grid cell, four times more than for finite volume methods.

	storage
flow field	$(3nm + n + 4)$R
geometry	84 R
topology	91 I
total	200 words

Table 1 Memory requirements per grid cell for the DG method. The following notations are used: n is the number of flow variables ($n = 5$), m the number of basis functions ($m = 4$), R refers to real variables (8 Bytes), I to integer variables (4 Bytes). Totals are in 8 Bytes words.

3 Computational complexity

In this section the computational complexity of the DG method is analyzed and compared to a well-tuned finite volume Jameson algorithm implemented in the multi-block structured flow solver ENFLOW (Ref. 3). Since the Jameson algorithm is optimal in terms of computational complexity, the reader should be aware that this is the strictest comparison possible.

The memory requirements of the above DG method are tabulated in Table 1. The memory is split into three parts: flow field, geometry (grid points, mass matrix, element integrals, etc.) and topology. The latter is required since the hexahedron grids are unstructured. The block-structured finite volume flow solver ENFLOW requires approximately 20 words per cell. The second order DG flow solver HEXADAP on unstructured meshes has four times more degrees of freedom and requires 2.5 times more memory per degree of freedom than the block-structured finite volume flow solver.

The number of floating point operations per grid cell per fine grid iteration (including coarse grid corrections)of HEXADAP is 21 kflop, that is, 5 kflop per degree of freedom per fine grid iteration. In Table 2 the main components of the computation and their respective work load is shown. The main part is the Osher flux difference scheme, followed closely by the slope limiter. Note that the solution of the moment equations constitutes 20 % of the work load. Certain geometric contributions are recomputed at each stage in the Runge-Kutta scheme, amounting to 10% of the work load. Note that since the face flux computations constitute about 50% of the work load, a four point quadrature rule for the flux evaluation would increase the total work load by a factor of 2.5. Hence the above approximations to the flux integrals significantly reduce the computational complexity. The single processor vector performance is 600 Mflop/s (30% peak) on average on a NEC SX-4 and is mainly bounded by memory access.

The finite volume flow solver ENFLOW requires 2 kflop per grid cell per fine grid iteration. The unstructured DG method has, however, four times more degrees of freedom and is 2.5 times more computationally expensive, per degree of freedom, than ENFLOW. In the next section it will be shown that the DG method is accurate on highly non-uniform grids, which require significantly less elements than structured grids.

	work load	average performance
Osher scheme	33 %	800
slope limiter	25 %	400
flow moments	20 %	1450
geometric contributions	10 %	400
left and right states	7 %	500
Runge Kutta	5 %	1500

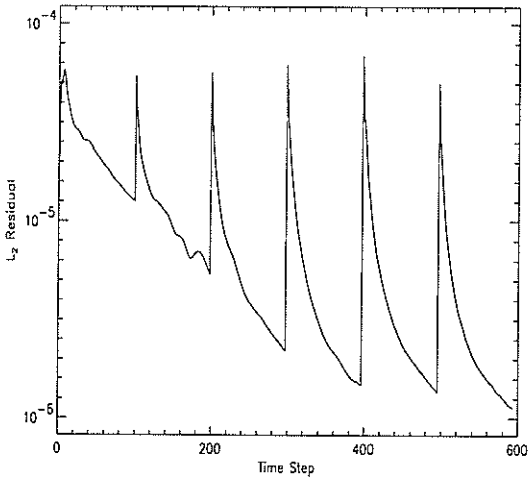
Table 2 Distribution of work and average single processor vector performance (in Mflop/s) in the DG method

4 Results

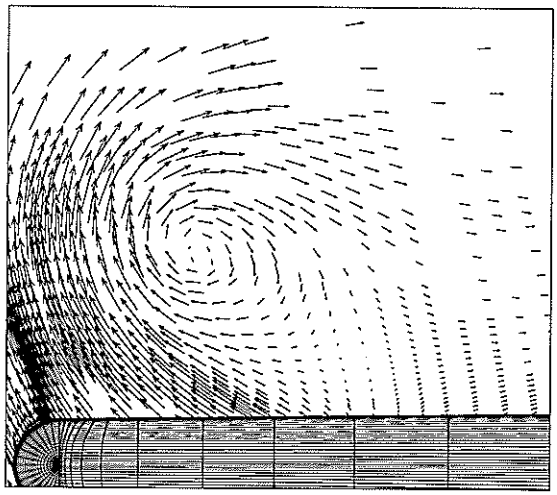
The balance between computational complexity and accuracy of the DG finite element method discussed in this paper is investigated by calculating the flow field about a generic wing at a free stream Mach number $M_\infty = 0.84$ and angle of attack $\alpha = 3.06°$. The wake vortices of the wing are difficult to capture over a large distance, especially if the grid is not aligned with the vortex core. After calculating an initial solution on a grid of 130,000 cells, the grid is adapted five times until a grid with 250,000 cells is obtained. After each adaptation the mesh is repartitioned for parallel load balance, and the flow is advanced for 100 multigrid cycles, see Figure 1(a). Grid refinement is only performed in the wake and not over the wing. The grid is refined at the vortex using a vortex sensor based on vortex strength and the total pressure loss. Note that the derivatives of the velocity are directly available in each cell, since the DG method has 20 degrees of freedom per cell, including a fully resolved gradient of the state vector.

Figure 1(b) shows the vortex in a cross-section at $x = 3$ (the wing tip is located at $x = 1.4$ and the wing span is three). In Figure 2 the vortex, shown as streamlines, at a cross-section one and half wing span behind the wing tip on the one time refined mesh is compared with the results on the final refined mesh. It can be clearly seen that the vortex is better resolved and extends further downstream. Figure 3 shows the final adapted grid at $x = 3$, $x = 6$, and $x = 9$. Note that the initial mesh is not aligned with the vortex core, but the grid refinement accurately captures the vortex.

Locally at the vortex the initially structured grid is refined twice in all three directions. In case of uniform refinement in, say, a quarter of the mesh, a structured grid with the same resolution would require at least $\frac{1}{4} \cdot 64 = 16$ times more grid points than the adapted grid (and would be difficult to generate since the vortex position is unknown beforehand). Hence the higher computational complexity of the DG method using locally refined meshes is compensated by its accuracy on non-uniform meshes.



(a) Convergence history



(b) Vortex at $x = 3$

Fig. 1 Multigrid convergence history of the L_2 residuals of the means and flow field of a generic wing ($M_\infty = 0.84$, $\alpha = 3.06°$). Only the residuals in the wake are measured.

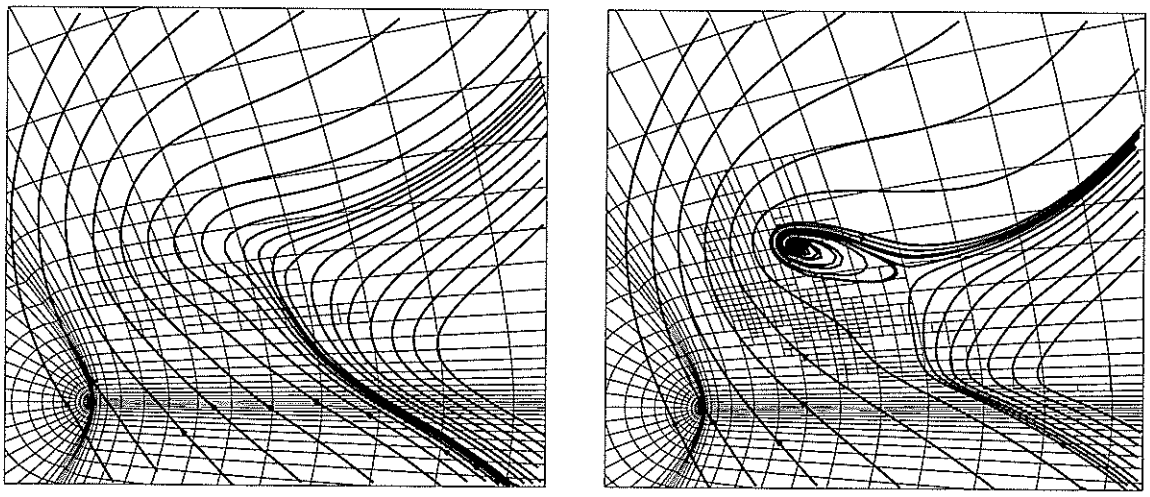
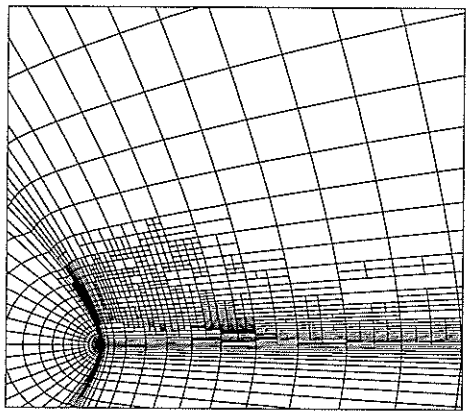
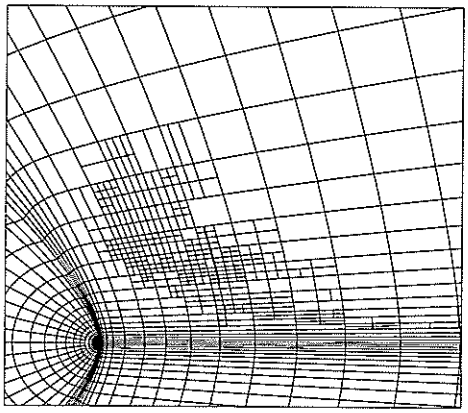


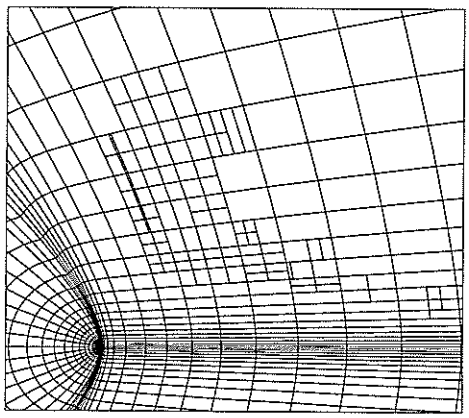
Fig. 2 Vortex comparison on one time refined grid (left) and final refined grid (right) at $x = 6$. The vortex is visualized using streamlines.



(a) mesh at $x = 3.0$



(b) mesh at $x = 6.0$



(c) mesh at $x = 9.0$

Fig. 3 Three cross-sections of the final refined non-uniform mesh behind a generic wing.

5 Conclusions

The DG method is efficient on highly non-uniform meshes and, combined with grid adaptation, is able to trace wake vortices over large distances. The computational complexity of the DG method per degree of freedom is 2.5 times the complexity of a finite volume block-structured method, both in flop count and memory use. This factor is similar to the difference between structured and unstructured (tetrahedra) finite volume schemes. The increased complexity of the unstructured DG method is compensated by its accuracy on non-uniform, locally refined, grids which require significantly less grid cells for the same resolution.

6 References

1. B. Cockburn, S. Hou and C.-W. Shu, The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case, *Math. Comput.* **54**, 545 (1990).
2. B. Cockburn and C.-W. Shu, The Runge-Kutta discontinuous Galerkin finite element method for conservation laws V: Multidimensional systems, *J. Comput. Phys.* **141**, 199 (1998).
3. J.C. Kok, J.W. Boerstoel, A. Kassies and S.P. Spekreijse, A robust multi-block Navier-Stokes flow solver for industrial applications, in Proceedings 3rd ECCOMAS CFD Conference, September 1996, John Wiley and Sons, Chichester, NLR TP 96323.
4. J.J.W. van der Vegt, Anisotropic grid refinement using an unstructured discontinuous Galerkin method for the three-dimensional Euler equations of gas dynamics, in *Proc. 12th AIAA CFD Conference, San Diego, California, 1995.* [AIAA Paper 95-1657-CP]
5. J.J.W. van der Vegt and H. van der Ven, Hexahedron based grid adaptation for future large eddy simulation, in *Proc. Progress and Challenges in CFD Methods and Algorithms, Seville, Spain, 1995.* [NLR TP 95514, AGARD CP-578, p. 22-1]
6. 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. Comput. Phys.* **141**, 46 (1998), NLR TP 97241.
7. H. van der Ven and J.J.W. van der Vegt, Partitioning and parallel development of an unstructured, adaptive flow solver on the NEC SX-4, in *Proc. Parallel Computational Fluid Dynamics '97 Conference, Manchester, England, 1997*, NLR TP 97329.
8. H. van der Ven and J.J.W. van der Vegt, Experiences with Advanced CFD Algorithms on NEC SX-4, in *Proc. Vector and Parallel Processing VECPAR '96*, edited by Palma and Dongarra, Lect. Notes in Computer Science, (Springer Verlag, 1997), NLR TP 96575.