NASA-CR-193659

GRANT
IN-08-CR
179799
9P

MCAT Institute
Progress Report
93-15

TURBULENCE MODELING OF FREE SHEAR LAYERS FOR HIGH-PERFORMANCE AIRCRAFT

Douglas L. Sondak

(NASA-CR-193659) TURBULENCE MODELING OF FREE SHEAR LAYERS FOR HIGH-PERFORMANCE AIRCRAFT Progress Report (MCAT Inst.) 9 p

N94-12799

Unclas

G3/08 0179799

August 1993

NCC2-663

MCAT Institute
3933 Blue Gum Drive
San Jose, CA 95127

CASI

Turbulence Modeling of Free Shear Layers for High-Performance Aircraft

Douglas L. Sondak

A35

Introduction

The High Performance Aircraft (HPA) Grand Challenge of the High Performance Computing and Communications (HPCC) program involves the computation of the flow over a high performance aircraft. A variety of free shear layers, including mixing layers over cavities, impinging jets, blown flaps, and exhaust plumes, may be encountered in such flowfields. Since these free shear layers are usually turbulent, appropriate turbulence models must be utilized in computations in order to accurately simulate these flow features.

The HPCC program is relying heavily on parallel computers. A Navier-Stokes solver (POVERFLOW) utilizing the Baldwin-Lomax algebraic turbulence model (Baldwin and Lomax 1978) has been developed and tested on a 128-node Intel iPSC/860 (Ryan and Weeratunga 1993). Algebraic turbulence models run very fast, and give good results for many flowfields. For complex flowfields such as those mentioned above, however, they are often inadequate. It was therefore deemed that a two-equation turbulence model will be required for the HPA computations.

In the present study, the $k-\epsilon$ two-equation turbulence model was implemented on the Intel iPSC/860. Both the Chien low-Reynolds-number model (Chien 1982) and a generalized wall-function formulation (Sondak and Pletcher 1993) have been included.

Milestones

Laminar Impinging Jet

The goal of the present study was to compute the flow and heat transfer for impinging turbulent axisymmetric jets. The parallel Navier-Stokes solver POVERFLOW had not previously been run for an axisymmetric geometry. A laminar axisymmetric jet flowfield was therefore computed on the iPSC/860, and the results were compared to those from an identical setup run on a Cray Y-MP (single preocessor) using OVERFLOW, the serial version of the code. Several coding bugs were found in POVERFLOW, and the two codes gave the same results after the bugs were rectified. The results were not compared with test data, since the purpose of the computation was simply to verify that the parallel and serial versions of the code gave the same results for axisymmetric geometries.

Parallel $k-\epsilon$ Model

A number of issues arise in the parallel implementation of two-equation turbulence models which are not of consideration for serial applications. Many two-equation models, including the Chien model, utilize exponential damping functions which are a function of distances to walls. In parallel codes, computing the distance to a wall will generally require message passing, since all of the grid points in a line directed away from a wall will not be stored in the same node. Since message passing is expensive, this is a disadvantage. Algebraic models also require distances to walls, so they also have the same disadvantage.

Table 1: Message Passing CPU Comparison

No. Nodes	PGE CPU (s)	PMP CPU (s)
1	1.142×10^{-2}	1.147×10^{-2}
2	1.004×10^{-2}	1.046×10^{-2}
4	1.116×10^{-2}	1.090×10^{-2}
8	1.235×10^{-2}	1.315×10^{-2}
16	1.345×10^{-2}	1.820×10^{-2}

Wall functions only require the distance to the grid line adjacent to the wall, so they have an advantage in this regard. The differences are quantified in the next section below.

Another issue for the parallel implementation is that of solvers. The $k-\epsilon$ model requires the solution of 2×2 block tridiagonal systems. The systems have two different forms, one for factors without source terms and one for factors with them. Since the forms of the blocks are known a priori, specialized solvers are used for improved performance. Pipelined Gaussian Elimination (PGE) was chosen for the turbulence model, since this method was shown to run fast for the Navier-Stokes solver (Ryan and Weeratunga 1993). An additional method was also investigated, in which messages were passed one plane at a time (PMP, planar message passing), rather than one point at a time as in PGE. This has the advantage of requiring fewer messages to be passed, with the disadvantage of additional idle time for some processors. To test the alternative method, a 2×2 block tridiagonal system containing 17 points was solved using both techniques. The results are shown in Table 1. Both methods give the same results for 1 node, since there is no message passing involved. For most cases, PGE is superior, so it was used in the final code.

Flat Plate Test Case

The computation of a 2-D turbulent boundary layer on a semi-infinite flat plate was computed as a test case for debugging purposes. A 91×91 grid was used, with 30 grid lines upstream of the leading edge. The Mach number was 0.2, and the Reynolds number at the outflow boundary, based on freestream velocity, was 9×10^6. Due to memory limitations on the iPSC/860, a minimum of 4 nodes was required for this case. Computations were done for two different coordinate orientations for debugging.

In comparing the serial and parallel runs, some small differences were noted. A laminar run was made, and this also showed the differences. The turbulence model gave identical results (approx. 14 significant figures) on the first step, and then small differences arose due to the differences in the Navier-Stokes portion of the code. Since the differences were small, and the Navier-Stokes solver has been previously, the accuracy was deemed adequate for some timing runs.

The flat plate test case was run for laminar flow, turbulent flow with wall functions, and turbulent flow with the Chien model. Timings are shown in Table 2. The Chien model requires 9–20% more CPU time than wall functions, depending upon the number of nodes used.

The wall function computations require 45–63% more CPU time than the laminar computations, the difference decreasing with increasing number of nodes. This result was disconcerting, since the wall function cases require about 40% more CPU time than laminar cases when using the serial code. In order to track down the cause of the increased difference, the Automated Instrumentation and Monitoring System (AIMS) (Yan et al. 1993) was used. AIMS is a set of tools which can be used to help monitor the performance of parallel codes. AIMS indicated an inordinate abount of time spent waiting for messages to arrive in the solvers. The problem is presently being investigated.

Table 2: Flat Plate Timings, μs/pt./step

No. Nodes	Laminar	Wall Functions	Chien
4	98.9	161.4	178.7
8	56.5	89.7	100.2
16	33.5	52.4	59.0
32	22.3	33.9	37.1
64	15.1	21.9	26.2

Literature Search

A literature search was carried out to find experimental data for axisymmetric turbulent impinging jet heat transfer. Although planar jet data is more plentiful, several prospective axisymmetric jet data sets have been identified (e.g., Sparrow and Lovell 1980, Hollworth and Gero 1984).

Present Status

There is a problem with running POVERFLOW in the batch mode. This is why no results have been presented for 128-node runs. The problem is being looked into by an Intel analyst.

The flat plate test case has been run to convergence. A program is being written to extract friction coefficients from the CFS solution files for wall-function cases. (This entails reading the solution file and running through the wall function routines.)

The Paragon computer should be on line soon. It is expected that some rewriting of POVERFLOW will be required for this machine, since the Paragon does not utilize a host computer.

An abstract for a paper, "Parallel Implementation of the $k-\epsilon$ Turbulence Model," has been submitted to AIAA for the 1994 Aerospace Sciences Meeting in Reno. The abstract is included in the appendix.

Customer Interactions

The $k-\epsilon$ subroutines have been transmitted to Daniel Barnette of Sandia National Laboratories (through appropriate channels). Consulting help has also been given to answer questions about the model, as well as questions about the F3D algorithm (he is using the subroutines in the F3D code).

I was contacted by Tom Austin of McDonnell Douglas Aerospace West in Long Beach. He is going to try to compute temperatures in the blown flaps of the C-17 transport, and had questions about turbulence modeling, algorithms, etc. We plan to remain in contact when he gets his computation going.

References

Baldwin, B. S., and Lomax, H. (1978). "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows." AIAA paper 78-257.

Chien, K.-Y. (1982). "Predictions of Channel and Boundary-Layer Flows with a Low-Reynolds-Number Turbulence Model." *AIAA Journal*, 20, 33–38.

Hollworth, B. R. and Gero, L. R. (1984). "Entrainment Effects on Impingement Heat Transfer. Part II: Local Heat Transfer Measurements." ASME paper 84-HT-20.

Ryan, J. S. and Weeratunga, S. K.. (1993). "Parallel Computation of 3-D Navier-Stokes Flowfields for Supersonic Vehicles." AIAA paper 93-0064.

Sondak, D. L. and Pletcher, R. H. (1993). "Application of Wall Functions to Generalized Nonorthogonal Curvilinear Coordinate Systems." AIAA paper 93-3107 (in preparation).

Sparrow, E. M. and Lovell, B. J. (1980). "Heat Transfer Characteristics of an Obliquely Impinging Jet." *ASME Journal of Heat Transfer*, **102**, 202–209.

Yan, J., Fineman, C., Hontalas, P., Schmidt, M., Listgarten, S., Mehra, P., Sarrukai, S., and Schulbach, C. (1993). "The Automated Instrumentation and Monitoring System (AIMS) Reference Manual." NASA Ames Research Center.

APPENDIX

Abstract for 32nd AIAA Aerospace Sciences Meeting
Jan. 10-13, 1994, Reno, NV

Parallel Implementation of the $k - \epsilon$ Turbulence Model

Douglas L. Sondak [*]
MCAT Institute, NASA Ames Research Center, Moffett Field, CA 94035-1000

Abstract

The $k-\epsilon$ turbulence model has been added to a Navier-Stokes solver on the Intel iPSC/860 massively parallel computer. Both the Chien low-Reynolds-number model and a high-Reynolds-number model with wall functions have been implemented. Flat plate and impinging jet flows have been computed using these models, as well as the Baldwin-Lomax algebraic turbulence model. Considerations specific to implementing these models on parallel machines are discussed, and timings for the various models are presented.

Introduction

Although the performance of supercomputers has been steadily improving, significant increases in computational speed are required before the analysis of complex viscous flows can be used as a routine design tool. An initiative has been established, the High Performance Computing and Communications (HPCC) program, to accelerate these improvements, primarily through the development of hardware and software for massively parallel computers. [1]

The HPCC program is divided into two areas, Computational AeroSciences (CAS) and Earth and Space Science (ESS). The CAS project contains four "grand challenge" problems, one of which is the High Performance Aircraft (HPA). The HPA grand challenge includes computations of the flowfield about a full aircraft undergoing a maneuver and of a powered lift aircraft in transition from hover to forward flight. [1] An appropriate turbulence model must be selected for these applications, taking into account issues of accuracy and computational efficiency. In the present study, the $k-\epsilon$ turbulence model has been added to an existing Navier-Stokes solver on the Intel iPSC/860. [2] Both the Chien[3] low-Reynolds-number $k-\epsilon$ model and the high-Reynolds-number model with wall functions [5] have been implemented. Computations and timings are presented for flow over a flat plate and for an impinging jet flow. In addition to the two versions of the $k-\epsilon$ model mentioned above, results of computations using the Baldwin-Lomax[4] algebraic turbulence model are also included for comparison. Relative computational speeds of the various models differ from those on serial computers, and the reasons are examined and discussed.

[*]Research Scientist, Member AIAA.

Intel iPSC/860

All computations in the present study have been carried out on the Intel iPSC/860 at NASA Ames Research Center. The iPSC/860 contains 128 processing nodes in a hypercube configuration. Each processing node consists of an Intel i860 processor and 8 megabytes of memory. The memory is dedicated to its respective node, and messages must be passed between nodes when data is to be shared. This message passing is time intensive, and one of the primary programming tasks is to minimize the amount of required message passing.

Turbulence Models

The three turbulence models employed in the present study are the $k-\epsilon$ model with wall functions,[5] the Chien low-Reynolds-number $k-\epsilon$ model, [3] and the Baldwin-Lomax[4] algebraic model.

$k-\epsilon$ Model

The nondimensional $k-\epsilon$ transport equations in transformed coordinates are given by

$$\frac{\partial \hat{Q}_t}{\partial \tau} + \frac{\partial \hat{E}_t}{\partial \xi} + \frac{\partial \hat{F}_t}{\partial \eta} + \frac{\partial \hat{G}_t}{\partial \zeta} - Re^{-1}\left(\frac{\partial \hat{E}_{tv}}{\partial \xi} + \frac{\partial \hat{F}_{tv}}{\partial \eta} + \frac{\partial \hat{G}_{tv}}{\partial \zeta}\right) = \hat{H}_t \qquad (1)$$

where the dependent variable vector is

$$\hat{Q}_t = J^{-1}\begin{bmatrix} \rho k \\ \rho \epsilon \end{bmatrix} \qquad (2)$$

The flux vectors are

$$\hat{E}_t = J^{-1}\begin{bmatrix} U\rho k \\ U\rho\epsilon \end{bmatrix} \tag{3}$$

$$\hat{F}_t = J^{-1}\begin{bmatrix} V\rho k \\ V\rho\epsilon \end{bmatrix} \tag{4}$$

$$\hat{G}_t = J^{-1}\begin{bmatrix} W\rho k \\ W\rho\epsilon \end{bmatrix} \tag{5}$$

$$\hat{E}_{tv} = J^{-1}\begin{bmatrix} \mathcal{N}_k\left(\alpha_1\frac{\partial\rho k}{\partial\xi} + \alpha_4\frac{\partial\rho k}{\partial\eta} + \alpha_5\frac{\partial\rho k}{\partial\zeta}\right) \\ \mathcal{N}_\epsilon\left(\alpha_1\frac{\partial\rho\epsilon}{\partial\xi} + \alpha_4\frac{\partial\rho\epsilon}{\partial\eta} + \alpha_5\frac{\partial\rho\epsilon}{\partial\zeta}\right) \end{bmatrix} \tag{6}$$

$$\hat{F}_{tv} = J^{-1}\begin{bmatrix} \mathcal{N}_k\left(\alpha_4\frac{\partial\rho k}{\partial\xi} + \alpha_2\frac{\partial\rho k}{\partial\eta} + \alpha_6\frac{\partial\rho k}{\partial\zeta}\right) \\ \mathcal{N}_\epsilon\left(\alpha_4\frac{\partial\rho\epsilon}{\partial\xi} + \alpha_2\frac{\partial\rho\epsilon}{\partial\eta} + \alpha_6\frac{\partial\rho\epsilon}{\partial\zeta}\right) \end{bmatrix} \tag{7}$$

$$\hat{G}_{tv} = J^{-1}\begin{bmatrix} \mathcal{N}_k\left(\alpha_5\frac{\partial\rho k}{\partial\xi} + \alpha_6\frac{\partial\rho k}{\partial\eta} + \alpha_3\frac{\partial\rho k}{\partial\zeta}\right) \\ \mathcal{N}_\epsilon\left(\alpha_5\frac{\partial\rho\epsilon}{\partial\xi} + \alpha_6\frac{\partial\rho\epsilon}{\partial\eta} + \alpha_3\frac{\partial\rho\epsilon}{\partial\zeta}\right) \end{bmatrix} \tag{8}$$

where

$$\mathcal{N}_k = \frac{1}{\rho}\left(\mu + \frac{\mu_t}{\sigma_k}\right) \tag{9}$$

$$\mathcal{N}_\epsilon = \frac{1}{\rho}\left(\mu + \frac{\mu_t}{\sigma_\epsilon}\right) \tag{10}$$

$$\alpha_1 = \xi_x^2 + \xi_y^2 + \xi_z^2 \tag{11}$$

$$\alpha_2 = \eta_x^2 + \eta_y^2 + \eta_z^2 \tag{12}$$

$$\alpha_3 = \zeta_x^2 + \zeta_y^2 + \zeta_z^2 \tag{13}$$

$$\alpha_4 = \xi_x\eta_x + \xi_y\eta_y + \xi_z\eta_z \tag{14}$$

$$\alpha_5 = \xi_x\zeta_x + \xi_y\zeta_y + \xi_z\zeta_z \tag{15}$$

$$\alpha_6 = \eta_x\zeta_x + \eta_y\zeta_y + \eta_z\zeta_z \tag{16}$$

and the source term vector is

$$\hat{H}_t = J^{-1}\begin{bmatrix} \mathcal{P} - \rho\epsilon \boxed{-Re^{-1}\frac{2\nu\rho k}{n^2}} \\ C_1\frac{\epsilon}{k}\mathcal{P} - C_2 f\frac{\rho\epsilon^2}{k}\boxed{-Re^{-1}\frac{2\nu\rho\epsilon}{n^2}e^{-C_4 n^+}} \end{bmatrix} \tag{17}$$

where

$$\mathcal{P} = Re^{-1}\left[\mu_t\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\delta_{ij}\frac{\partial u_k}{\partial x_k}\right)\right]\frac{\partial u_i}{\partial x_j} - \frac{2}{3}\rho k\frac{\partial u_k}{\partial x_k} \tag{18}$$

$$n^+ = Re^{-1}\frac{u_* n}{\nu} \tag{19}$$

$$f = 1\boxed{-\frac{0.4}{1.8}e^{-\left(Re\frac{k^2}{6\nu\epsilon}\right)^2}} \tag{20}$$

and

$$\mu_t = Re\ C_\mu\rho\frac{k^2}{\epsilon}\left(1\boxed{-e^{-C_3 n^+}}\right) \tag{21}$$

Here, n is the normal distance from the wall.

Table 1: Constants for two-equation models

	C_1	C_2	C_3	C_μ	σ_k	σ_ϵ
Chien low-Re[3]	1.35	1.8	0.0115	0.09	1.0	1.3
high-Re[7]	1.44	1.92	-	0.09	1.0	1.3

The transport equations for the Chien low-Reynolds-number model are the same as those for the high-Reynolds-number model with the addition of terms to account for the effects of solid walls. The low-Reynolds-number terms are shown in boxes in the above equations. The standard set of constants, shown in table 1, are used for all of the present computations.

Although the present application involves simple geometries, it was desired to make the parallel code as general as possible. The wall function formulation used in the present study[5] may be used for completely general geometries for flows over walls with heat flux.

Baldwin-Lomax Model

The Baldwin-Lomax[4] turbulence model is a two-layer algebraic model. The turbulent viscosity is given by

$$\mu_t = \begin{cases} (\mu_t)_{inner} & n \le n_{crossover} \\ (\mu_t)_{outer} & n > n_{crossover} \end{cases} \tag{22}$$

where $(\mu_t)_{inner}$ and $(\mu_t)_{outer}$ are functions of local flow variables and n is the normal distance from the wall. The important point here is that the turbulent viscosity is a function of distance from the wall. This has implications for the parallel implementation which do not arise in serial codes, as will be discussed in the next section.

Parallel Considerations

Since the $k-\epsilon$ model requires the solution of two transport equations in addition to the five Navier-Stokes equations, one would expect the turbulent solutions to require approximately 40% more time than laminar solutions, and this has been observed in practice with serial codes. The Baldwin-Lomax model is much simpler, consisting of algebraic equations, and it requires approximately 5% more time than laminar solutions on serial machines.

Some turbulence models require computation of normal distances to walls. This is true for the low-Reynolds-number terms in equations 17 and 21 and in the Baldwin-Lomax model, equation 22. This is a disadvantage for parallel implementations, since the locations of the walls must be passed to other processors. Distances to walls are not required for the $k-\epsilon$ model if wall functions are employed. One of the purposes of the present study is to quantify the effects of these differences on computational speed.

Domain Decomposition

The computational grid is divided into subdomains, and each subdomain is assigned to one processor. At subdomain surfaces which are internal to the overall domain, one plane of data adjacent to each internal surface is required for the computation of the right hand side of each set of equations. This data may be obtained by passing messages between subdomains, or an additional, overlapping plane may be included at each internal subdomain surface. The latter approach requires additional memory to store data for the overlapping planes, but runs faster, since message passing is avoided. This is the approach taken in the present code. Additional details of the domain decomposition are given by Ryan and Weeratunga.[2]

Solvers

The Navier-Stokes and $k-\epsilon$ equations are solved loosely coupled. A second-order, diagonalized approximate-factorization scheme (ARC3D) [6] is employed for the Navier-Stokes equations, and a first-order upwind scheme with a block tridiagonal solver is used for the $k-\epsilon$ equations. Both equation sets are solved using pipelined Gaussian elimination. Pipelining refers to the fact that short messages are passed across subdomain boundaries, one grid point at a time, as soon as they are available.

Results

A simple test case was desired which could be used to verify that the parallel implementation of the turbulence model was working correctly, and to obtain some preliminary timings. Turbulent, subsonic flow over a flat plate was chosen for this purpose.

One of the eventual goals of the present work is to compute the heat transfer and flowfield for the impinging jets of a powered lift aircraft in hover. Toward this end, the second test case is an axisymmetric jet impinging normally on a flat plate.

Flat Plate

Flow over a semi-infinite flat plate was solved using a 91×91 grid. The grid is clustered at the plate surface and at the leading edge, as shown in figure 1.

[Dummy figures are enclosed for:
figure 2, friction coefficient along plate
table 2, CPU/grid pt./iteration
figure 3, parallel efficiency]

Impinging Jet

The second test case was an impinging jet impinging on a hot surface. The 59×101 grid is shown in figure 4.

[Dummy figures are enclosed for:
figure 5, friction coefficient along plate
table 3, CPU/grid pt./iteration
figure 6, parallel efficiency]

Conclusions and Recommendations

[Recommendations of appropriate turbulence models for parallel computations of the the impinging jet problem will be made based on timings and accuracy of the Nusselt number distributions.]

References

[1] Holst, T. L., Salas, M. D., and Claus, R. W. (1992). "The NASA Computational Aerosciences Program — Toward Terraflops Computing." AIAA-92-0558.

[2] Ryan, J. S. and Weeratunga, S. K. (1993). "Parallel Computation of 3-D Navier-Stokes Flowfields for Supersonic Vehicles." AIAA-93-0064.

[3] Chien, K.-Y. (1982). "Predictions of Channel and Boundary-Layer Flows with a Low-Reynolds-Number Turbulence Model." *AIAA Journal*, **20**, 33–38.

[4] Baldwin, B. S. and Lomax, H. (1978). "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows." AIAA-78-257.

[5] Sondak, D. L. and Pletcher, R. H. (1993). "Application of Wall Functions to Generalized Nonorthogonal Curvilinear Coordinate Syatems." AIAA-93-3107.

[6] Pulliam, T. H. and Chaussee, D. S. (1981). "A Diagonal Form of an Implicit Approximate-Factorization Algorithm." *Journal of Computational Physics*, **39**, No. 2, 347–363.

[7] Launder, B. E. and Spalding, D. B. (1974). "The Numerical Computation of Turbulent Flows." *Computer Methods in Applied Mechanics and Engineering*, **3**, 269–289.