MCAT Institute
Final Report
94-12

Turbulence Modeling of Free Shear Layers For High Performance Aircraft

Douglas Sondak

October 1993

NCC2-663

MCAT Institute
3933 Blue Gum Drive
San Jose, CA 95127

Turbulence Modeling of Free Shear Layers For High Performance Aircraft

Douglas Sondak

In many flowfield computations, accuracy of the turbulence model employed is frequently a limiting factor in the overall accuracy of the computation. This is particularly true for complex flowfields such as those around full aircraft configurations. Free shear layers such as wakes, impinging jets, (in V/STOL applications), and mixing layers over cavities are often part of these flowfields.

Although flowfields have been computed for full aircraft, the memory and CPU requirements for these computations are often excessive. Additional computer power is required for multi-disciplinary computations such as coupled fluid dynamics and conduction heat transfer analysis. Massively parallel computers show promise in alleviating this situation, and the purpose of this effort was to adapt and optimize CFD codes to these new machines.

The objective of this research effort was to compute the flowfield and heat transfer for a two-dimensional jet impinging normally on a cool plate. The results of this research effort were summarized in an AIAA Paper titled "Parallel Implementation of the k-e Turbulence Model". Appendix A contains the full paper.

APPENDIX - A

Parallel Implementation of the κ–ε Turbulence Model

Douglas L. Sondak
MCAT Institute
Moffett Field
California

Parallel Implementation of the k-ϵ Turbulence Model

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

Abstract

The k-ϵ turbulence model has been added to a parallel Navier-Stokes solver on an Intel iPSC/860 massively parallel computer. Both a high-Reynolds-number model with wall functions and the Chien low-Reynolds-number model have been implemented. Two flowfields have been computed: flow over a flat plate, and the flow and heat transfer for a two-dimensional jet impinging normally on a cool plate. Considerations specific to implementing these models on parallel machines are discussed, and timings for the two models are presented.

Notation

Roman Symbols

C_μ	constant for $k-\epsilon$ model (see Table 1)
C_f	friction coefficient
C_1	constant for $k-\epsilon$ model (see Table 1)
C_2	constant for $k-\epsilon$ model (see Table 1)
C_p	specific heat at constant pressure
H_t	source term vector for $k-\epsilon$ equations
J	Jacobian of coordinate transformation
k	turbulent kinetic energy
L	reference length
n	normal distance from wall
$\mathcal{P}$	rate of production of turbulent kinetic energy
Pr	Prandtl number
Q	dependent variable vector
Re	Reynolds number, $Re = a_\infty L/\nu_\infty$
Re_θ	Reynolds number based on freestream velocity and momentum thickness
St	Stanton number, $St = q_w/\rho C_p T_w U_j$
T	temperature
T^+	temperature in wall coordinates, $T^+ = (T - T_w)\,\rho C_p u_* / q_w$
u, v, w	velocity components in Cartesian coordinate directions
u^+	velocity normalized by friction velocity
u_*	friction velocity
U, V, W	contravariant velocity components
U_j	jet velocity
x, y, z	Cartesian coordinates
y^+	distance from wall in wall coordinates, $y^+ = u_* n/\nu$

Greek Symbols

γ	ratio of specific heats
δ_{ij}	Kronecker delta
ϵ	dissipation rate of turbulent kinetic energy
κ	von Karman constant, 0.41
λ	$\sqrt{2/C_f}$
μ	molecular viscosity
μ_t	turbulent viscosity
ν	kinematic viscosity
ξ, η, ζ, τ	transformed coordinates
ρ	density
σ_ϵ	constant for $k-\epsilon$ model, 1.3
σ_k	constant for $k-\epsilon$ model, 1.0
τ	shear stress; transformed time in eqn. (1)

Subscripts and Superscripts

i, j, k, l	tensor indices
t	turbulent
v	viscous
w	wall
x, y, z	partial differentiation in Cartesian coordinate directions
∞	freestream condition

Other Symbols

$(\hat{\ })$	vector in transformed coordinates
μs	microseconds

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

*Research Scientist, Member AIAA. Presently at United Technologies Research Center, East Hartford, CT. 06108

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. Towards this end, the k-ϵ turbulence model has been added to an existing Navier-Stokes solver[2] on an Intel iPSC/860. Both the Chien low-Reynolds-number k-ϵ model[3] and a high-Reynolds-number k-ϵ model[4] with wall functions[5] have been implemented. Flow over a flat plate has been computed in order to test the models and to get an initial set of CPU times. To test the models on a more realistic geometry which is applicable to the HPA grand challenge problem, the flow and heat transfer for a 2D planar jet impinging on a cool surface was then computed. Relative computational speeds of the two turbulence models differ from those on serial computers, and the causes of the differences are examined and discussed.

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 a 40MHz Intel i860 processor and 8MB of memory. Input/output is handled by 10 additional nodes (386-based processors), each of which has 8MB of memory. Ten 760MB disks are available for file storage. The front-end machine is an Intel computer with an 80386 processor and 8MB of memory.

On the iPSC/860, the memory for each processing node is dedicated to that node, and messages must be passed between nodes when data are to be shared. Message passing is costly in CPU time, and one of the primary programming tasks is to minimize this cost through careful choice of domain decomposition and the use of appropriate solvers.

k-ϵ Model

The nondimensional k-ϵ transport equations in transformed coordinates are given by

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

where the dependent variable vector is

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

The flux vectors are

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

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

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

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

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

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

where

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

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

$$\alpha_1 = \xi_x^2 + \xi_y^2 + \xi_z^2 \quad (11)$$

$$\alpha_2 = \eta_x^2 + \eta_y^2 + \eta_z^2 \quad (12)$$

$$\alpha_3 = \zeta_x^2 + \zeta_y^2 + \zeta_z^2 \quad (13)$$

$$\alpha_4 = \xi_x\eta_x + \xi_y\eta_y + \xi_z\eta_z \quad (14)$$

$$\alpha_5 = \xi_x\zeta_x + \xi_y\zeta_y + \xi_z\zeta_z \quad (15)$$

$$\alpha_6 = \eta_x\zeta_x + \eta_y\zeta_y + \eta_z\zeta_z. \quad (16)$$

The source term vector is

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

where

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

Table 1: Constants for $k-\epsilon$ models

	C_1	C_2	C_3	C_4	C_μ	σ_k	σ_ϵ
Chien low-Re[3]	1.35	1.8	0.0115	0.5	0.09	1.0	1.3
high-Re[4]	1.44	1.92	-	-	0.09	1.0	1.3

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

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

and

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

Here, n is the normal distance from the wall.

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, is used for all of the present computations.

Although the present application involves simple geometries, the parallel code was made as general as possible. A wall function formulation[5] which is applicable to general geometries with wall heat transfer was therefore employed.

Timing Considerations

In a previous study,[5] 2D cases were computed on a Cray Y-MP using the present turbulence models. In those computations, a block Beam-Warming algorithm was utilized, which differs from the present diagonalized algorithm, but the relative timings for the turbulence models are still valid. The resulting Y-MP timings were as follows: laminar, 35.7 μs/pt./step; k-ϵ with wall functions, 49.5 μs/pt./step; and Chien model, 50.9 μs/pt./step. The wall function and Chien cases required 38.7% and 42.6% more time respectively compared to the laminar computations. The Chien model requires slightly more time than wall functions (2.8%) due to additional terms in the transport equations and the source term Jacobians.

Some turbulence models require computation of normal distances to walls. This is true for the low-Reynolds-number terms in equations (17) and (21). This is a disadvantage for MIMD implementations, since the locations of the walls must be passed to other nodes. Distances to walls are not required for the k-ϵ 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.

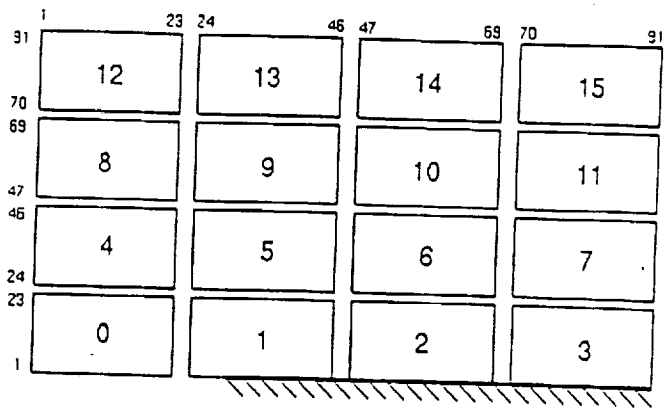


Figure 1: Typical domain decomposition

Domain Decomposition

The computational grid is divided into subdomains, and each subdomain is assigned to one node. 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. These data may be obtained by passing messages between subdomains, or alternatively, 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 some message passing is avoided. This is the approach taken in the present code.

A typical domain decomposition, for a flat plate computation utilizing 16 nodes, is shown in Fig. 1. In this figure, the ξ direction is parallel to the plate and the η direction is normal to the plate. Subdomains are represented by boxes, and the corresponding node number is shown in each subdomain. The ranges of grid lines contained in the nodes are shown on the left side and top of the figure. As an example of the grid overlap, data for $\xi = 23$ through $\xi = 47$ are stored on node 5, even though the solution domain for that node is $\xi = 24$ through $\xi = 46$. Data for $\xi = 46$ through $\xi = 70$ are stored on node 6, etc. Additional details of the domain decomposition are given by Ryan and Weeratunga.[2]

Solvers

The Navier-Stokes and k-ϵ equations are solved loosely coupled. A second-order, diagonalized, approximate-

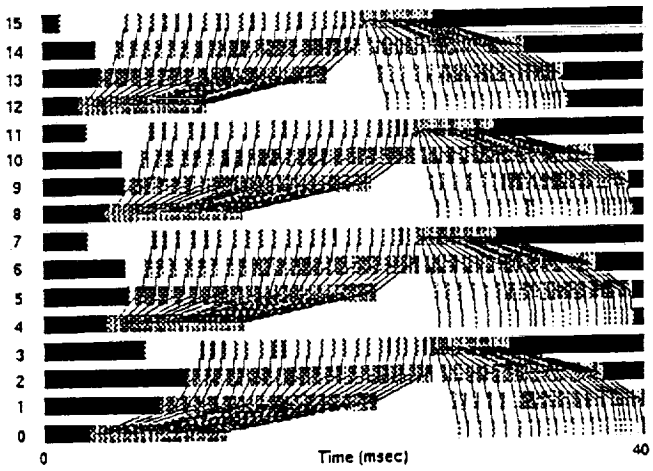


Figure 2: AIMS diagram for scalar tridiagonal solver

factorization scheme (ARC3D)[6] is employed for the Navier-Stokes equations, and a first-order upwind scheme with block tridiagonal solvers is used for the k-ϵ equations. For the k-ϵ equations, the source terms are linearized, and the source term Jacobian is arbitrarily included in a single factor. These equations are coupled to one another solely through the source terms, so the equations in the two factors which do not include the source term Jacobians are uncoupled from one another. For these factors, a scalar tridiagonal solver is employed. For the factor containing the source term Jacobian, a 2×2 block tridiagonal solver is employed.

Both equation sets are solved using Pipelined Gaussian Elimination (PGE), since this technique was found to be a resonable compromise between CPU time and memory requirements by Ryan and Weeratunga[2] for the Navier-Stokes solver. In PGE, a short message is passed across the subdomain boundary for each grid point as soon as the data are available.

A tool has been developed at NASA Ames, the Automated Instrumentation and Monitoring System (AIMS),[7] to help visualize the operation of parallel programs. AIMS was utilized to verify the coding of the k-ϵ solvers. An example AIMS diagram for the scalar tridiagonal solver for the domain decomposition of Fig. 1 is shown in Fig. 2. Each horizontal bar represents a processor, numbered on the left side of the figure, and the abcissa represents time. Processors are in operation in times during which the bars are solid, and are idle at other times, typically waiting for messages to arrive. Lines connecting processor bars represent messages being passed. The gray bars represent computation time spent in the solver, and the black bars represent computation time in other subroutines. For this configuration of processors, the forward and backward sweeps are clearly shown (e.g., processors $0 - 1 - 2 - 3 - 2 - 1 - 0$). The load imbalance (idle time) which is inherent in PGE is shown by the extent of the blank areas.

Wall Functions

For the present study, the wall function formulation of Sondak and Pletcher[5] has been extended to include flows with wall heat transfer. For each grid point adjacent to a wall, it is determined whether the point lies in the viscous sublayer or in the log region (neglecting the buffer region) by computing the value of y^+. If the point is in the viscous sublayer, the heat transfer rate is computed from

$$q_i = -\frac{\gamma \mu}{Pr} \frac{\partial e}{\partial x_i}, \tag{22}$$

assuming constant specific heats.

For points in the log region, the temperature log equation[8] is employed:

$$T^+ = \frac{1}{\kappa} \ln y^+ + A \tag{23}$$

where $A = 4.6$ for the standard value $Pr_t = 0.9$. The wall heat transfer rate is then given by

$$q_w = \frac{(T - T_w)\, \rho C_p u_*}{T^+}. \tag{24}$$

The direction of heat transfer is normal to the wall, and the magnitude is equal to q_w, so the heat transfer vector is completely specified. The heat transfer vector is then transformed to the Cartesian coordinate system. The Cartesian components are then substituted into the right hand side of the energy equation. This method is analogous to that used for shear stress by Sondak and Pletcher.[5]

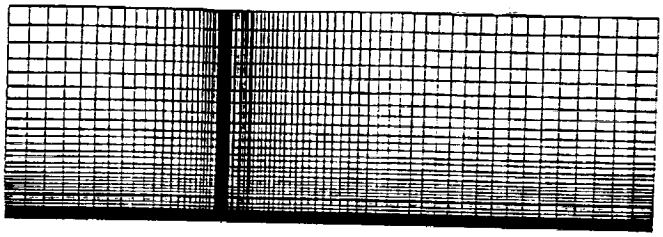
Results

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.

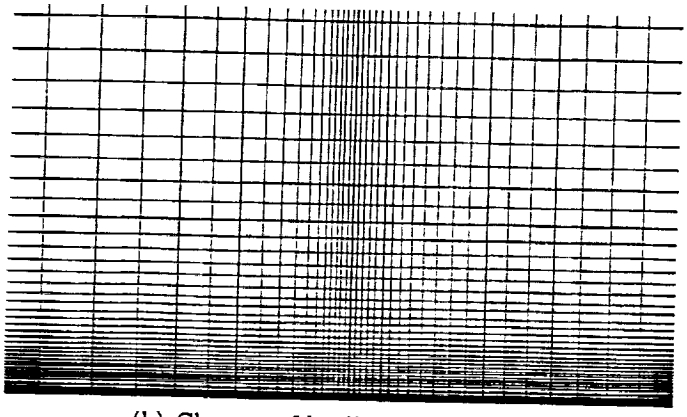
One of the eventual goals of the present work is to compute the heat transfer and flowfield for the impinging jets of a powered lift aircraft in hover for the HPA grand challenge. Toward this end, the second test case is the flow and heat transfer for a jet impinging normally on a flat plate.

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



(a) Whole grid



(b) Closeup of leading edge region

Figure 3: Flat plate grid

at the leading edge, as shown in Fig. 3. A grid spacing of 1×10^{-5} (normalized by the plate length) at the wall was chosen to yield $y_w^+ \approx 3$, keeping the grid point adjacent to the wall in the viscous sublayer, as required by the Chien model. There are 30 grid lines upstream of the leading edge, and 61 on the plate.

A value of $M_\infty = 0.2$ was chosen to maintain a reasonable convergence rate with essentially incompressible flow. The Reynolds number based on freestream velocity was approximately equal to 9×10^6 at the outflow boundary. A small value of freestream turbulent kinetic energy was chosen ($k/u_\infty^2 = 0.00025$) based on past experience,[5] and the freestream dissipation rate was chosen such that $\mu_{t_\infty} = 1$. At the inflow boundary, density, momentum components, k, and ϵ were set to freestream values, and pressure was extrapolated. At the outflow boundary, pressure was fixed at the freestream value, and density, momentum components, and k and ϵ were extrapolated. On the stagnation streamline, density, streamwise momentum, pressure, k, and ϵ were extrapolated and normal momentum was set to zero (inviscid wall). On the plate, density and pressure were extrapolated, and momentum components were set to zero (no-slip). For the Chien model, k and ϵ were set to zero at the plate. For wall functions, k and ϵ at the plate are part of the formulation. At the outer edge, all quantities were extrapolated.

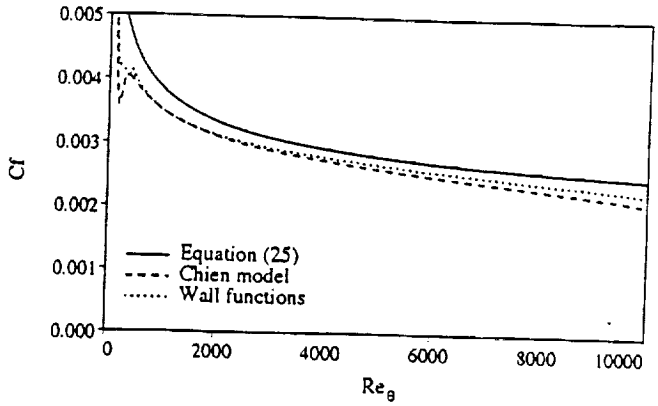


Figure 4: Flat plate friction coefficient

Table 2: Flat plate timings

	cpu time (μs/pt./step)		
no. nodes	laminar	wall fcn.	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

A semiempirical equation for friction coefficient as a function of momentum-thickness Reynolds number is given by[8]

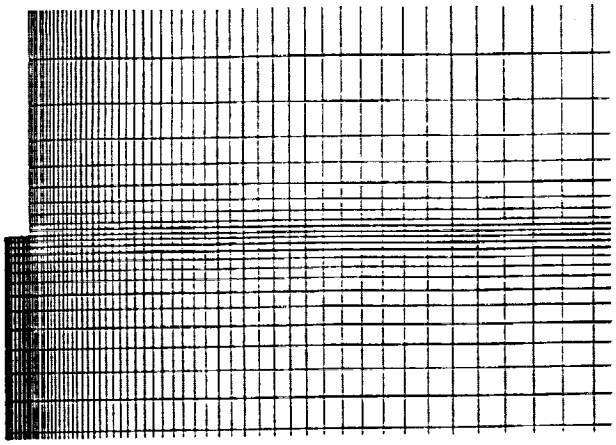
$$Re_\theta = \left(3.75 - \frac{24.778}{\lambda}\right) e^{0.4(\lambda - 8)} \qquad (25)$$

where $\lambda = \sqrt{2/C_f}$. The computed friction coefficient distributions are compared with this equation in Fig. 4.

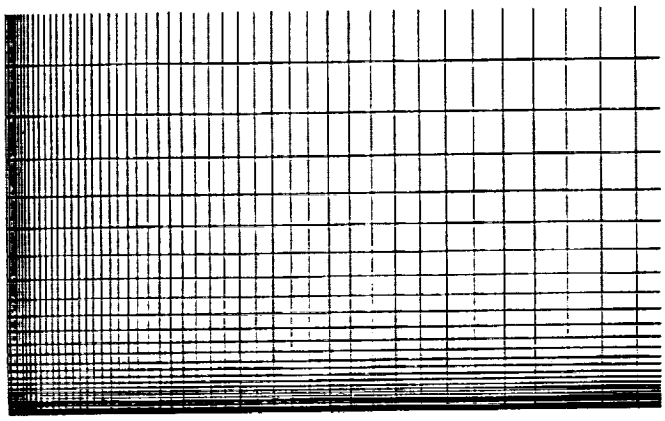
Table 2 shows CPU times for both turbulence models as well as laminar flow. The Chien model requires 10-20% more CPU time than the high-Reynolds-number model with wall functions. In the serial code discussed above, the the Chien model only required 2.8% more time. This difference is attributable to the additional message passing required by the Chien model for computation of the damping functions.

Impinging Jet

The second test case was a slot jet impinging on a cool constant-temperature surface, as studied experimentally by Schauer and Eustis.[9] The configuration chosen for the present computation had a height-to-jet-width ratio of 40, a jet Reynolds number of 4.08×10^4, and a jet-to-wall-



(a) Nozzle region



(b) Impingment region

Figure 5: Impinging jet grid

temperature ratio of 1.105. The ambient temperature was equal to the jet temperature.

Parts of the 121 × 121 grid is shown in Fig. 5. The grid spacing at the wall was set to 2×10^{-4} for a value of $y_w^+ \approx 1$. A grid with a wall spacing of 1×10^{-3} ($y_w^+ \approx 5$) was tried initially, but it did not have sufficient resolution for the thermal boundary layer. The domain is 60 slot widths wide and 60 slot widths high. The boundary conditions are as follows: Symmetry is imposed on all quantities at the jet centerline. On the surface of the channel from which the jet discharges, density and pressure are extrapolated, and momentum components are set to zero (no-slip). For the Chien model, the damping functions are not applied at this surface in order to minimize message-passing. On the plate, momentum components are set to zero (no-slip), the temperature is fixed, and pressure is extrapolated. For the Chien model, k and ϵ are set to zero at the plate. For wall functions, k and ϵ at the plate are part of the wall function formulation. At

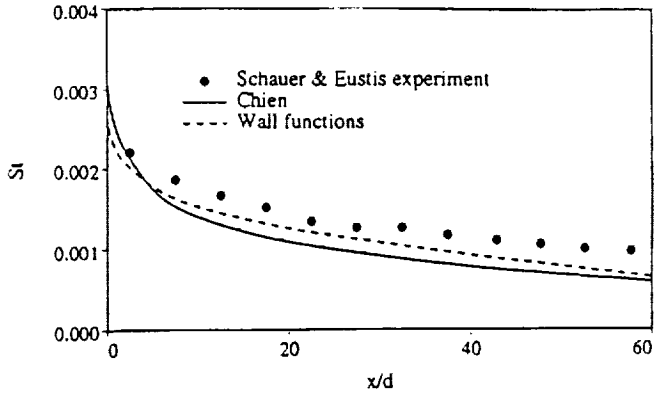


Figure 6: Impinging jet heat transfer

Table 3: Impinging jet timings

	cpu time (μs/pt./step)		
no. nodes	laminar	wall fcn.	Chien
8	52.7	86.7	95.2
16	33.0	52.3	58.6
32	18.8	29.8	32.9
64	11.8	18.6	22.6

the outflow boundary, pressure is fixed to the freestream value, and density, momentum components, k, and ϵ are extrapolated. At the outer edge, all quantities are extrapolated.

The computed flowfield was unsteady, with vortices rolling up at the edge of the jet and traversing the domain. The period of the unsteadiness was determined by monitoring the temperature of the flow at a grid point adjacent to the plate in the wall jet, and the resulting Strouhal number was equal to 0.29. The heat transfer rates were determined by averaging the values over every time step in one period (about 700 steps). The resulting Stanton number distributions are compared with the experimental data in Fig. 6. It should be noted that the grid was fine enough that the inner layer of the two-layer wall function formulation was called into play at all points. The temperature log equation could not be tested for the present configuration due to the thermal boundary layer resolution issue mentioned above.

A table of timings for the impinging jet case is shown in Table 3. The trends are the same as those for the flat plate case, as is to be expected. No timings are shown for 4 nodes because there was not sufficient memory available for the 121 × 121 grid.

Summary and Conclusions

The $k-\epsilon$ turbulence model has been added to a parallel Navier-Stokes solver on an Intel iPSC/860. Two test

cases have been computed: flow over an adiabatic flat plate, and an impinging jet flow with wall heat transfer. Two wall treatments were used for the $k-\epsilon$ model, wall functions, and the Chien low-Reynolds-number model.

The Chien model incurred an average CPU penalty of 13% compared to the wall function computations (on the same grids). Approximately 10% is attributable to message passing required in the calculation of damping functions. The remaining 3% was also seen in a serial implementation, and is therefore not a function of message passing.

Some low-Reynolds-number models, such as that of Jones and Launder,[10] do not utilize damping functions. These models therefore will have an advantage in terms of CPU time for parallel applications.

The impinging jet Stanton number distribution looks somewhat better for wall functions than for the Chien model, but it should not be concluded that wall functions will generally yield better results for this type of flowfield. Applications which exercise the temperature log equation part of the wall function formulation must be computed before any conclusions can be reached.

After additional validation, the next step will be to couple the present code with a thermal conduction solver[11] to study coupled thermal conduction/convection problems.

Acknowledgement

Special thanks go to Rob Van der Wijngaart for his helpful discussions on parallel processing issues (and on ways to solve the world's problems).

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

[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] Yan, J. C. (1994). preprint of "Performance Tuning with AIMS - An Automated Instrumentation and Monitoring System for Multicomputers." Proceedings of the 27th Hawaii International Conference on System Sciences, Kona, Hawaii, January, 4th - 7th, 1994.

[8] White, F. M. (1974). *Viscous Fluid Flow.* New York: McGraw-Hill.

[9] Schauer, J. J. and Eustis, R. H. (1963). *The Flow Development and Heat Transfer Characteristics of Plane Turbulent Impinging Jets.* Stanford: Stanford University Dept. of Mechanical Engineering. Technical Report No. 3.

[10] Jones, W. P. and Launder B. E. (1972). "The Prediction of Laminarization with a Two-Equation Model of Turbulence." *Int. J. Heat Mass Transfer.* 15, 301-314.

[11] Van der Wijngaart, R. F. (1993). preprint of "Efficient Implementation of a 3-Dimensional ADI Method on the iPSC/860." to be presented at Supercomputing '93, Portland, OR, Nov. 15 - 19, 1993.