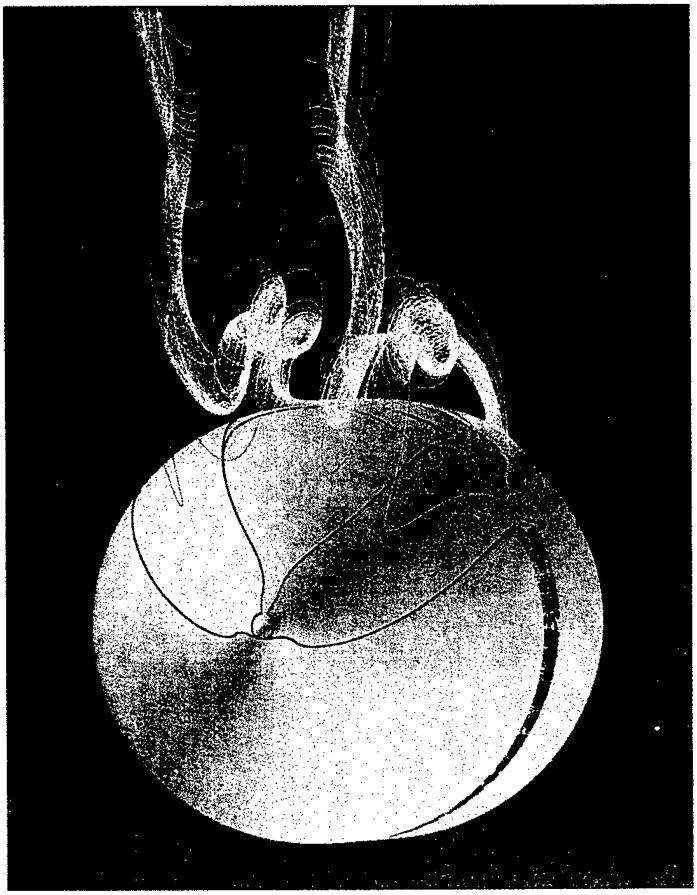
Naval Surface Warfare Center
Carderock Division

West Bethesda, Maryland 20817-5700

CRDKNSWC/HD-1318-02 Nov 1997

Hydromechanics Directorate
Research and Development Report

TRANSITION AND TESTING AT CDNSWC OF THE UNCLE.6DOF SUBMARINE MANEUVERING CODE



By

H.J. Haussling

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	November 1997	R & D Final

4. TITLE AND SUBTITLE
Transition and Testing at CDNSWC of the UNCLE.6DOF Submarine Maneuvering Code

5. FUNDING NUMBERS
PE: 0602121N
TASK: R2132-MS2
WU: 5060-741

6. AUTHOR(S)
Henry J. Haussling

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Surface Warfare Center Carderock Division
Hydromechanics Directorate, Code 5400
9500 MacArthur Boulevard
West Bethesda, MD 20817-5700

8. PERFORMING ORGANIZATION REPORT NUMBER
CRDKNSWC/HD-1318-02

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Ship Structures and Systems S&T Division (334)
Office of Naval Research
800 North Quincy St.
Arlington, VA 22217-5000

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release, distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The UNCLE.6DOF submarine maneuvering code, developed by Mississippi State University and The Pennsylvania State University, has been transferred to Carderock Division, Naval Surface Warfare Center and installed at the Navy Hydrodynamic/Hydroacoustic Technology Center. The code has been exercised on various test cases and results have been compared with measured data. Steady and unsteady, straight ahead and turning trajectories of appended and unappended bodies have been considered. The power and generality of the software are demonstrated and recommendations for further development are discussed.

14. SUBJECT TERMS
Reynolds-averaged Navier-Stokes equations, six degrees of freedom, submarine hydrodynamics, maneuvering

15. NUMBER OF PAGES
ii + 60

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	Same as Report

CONTENTS

FIGURES

ABSTRACT

The UNCLE.6DOF submarine maneuvering code, developed by Mississippi State University and The Pennsylvania State University, has been transferred to Carderock Division, Naval Surface Warfare Center and installed at the Navy Hydrodynamic/Hydroacoustic Technology Center. The code has been exercised on various test cases and results have been compared with measured data. Steady and unsteady, straight ahead and turning trajectories of appended and unappended bodies have been considered. The power and generality of the software are demonstrated and recommendations for further development are discussed.

ADMINISTRATIVE INFORMATION

The work was sponsored by the Office of Naval Research, Ship Structures and Systems S&T Division (Code 334), under the Maneuvering and Control Task of the FY97 Submarine Hull, Mechanical, and Electrical Technology Program (PE0602121N). The work described in this report was performed by the Propulsor Department of the Hydromechanics Directorate, Carderock Division, Naval Surface Warfare Center.

This report is submitted in partial fulfillment of the research in the Maneuvering and Control Task (R2132-MS2) under Milestone 3, Mississippi Code Transition, of Subtask 1, Maneuvering Prediction and Validation. The work was funded under Work Unit Number 1-5060-741-40.

INTRODUCTION

A research team consisting of personnel from the Engineering Research Center at Mississippi State University (MSU) and the Applied Research Laboratory at The Pennsylvania State University (PSU) developed a physics-based method for predicting the forces and moments and the resulting motion of a maneuvering underwater vehicle.[1] The approach taken was to couple a version of the UNCLE code for the numerical solution of the Reynolds-averaged Navier-Stokes (RANS) equations[2] with software for integrating the six degree of freedom (6DOF) equations governing a vehicle's motion. This latter software was derived directly from the TRJv code.[3] In the resulting system, referred to here as UNCLE.6DOF, the RANS portion is used to advance the flow field in time and compute the instantaneous forces on the vehicle. The 6DOF routines are then used to reposition the body in response to those forces. This two-part sequence is repeated cyclicly to construct a vehicle trajectory. Numerous test cases and examples were presented.[1]

There is a tremendous potential of such submarine maneuvering software for supplementing existing, more simplified, prediction tools. For this reason, Carderock Division, Naval Surface Warfare Center (CDNSWC) has accepted the task of transitioning the

code to the Navy and, in the process, contributing to the testing and validation begun, and still underway, at MSU and PSU. This transition was started in FY96 by J.A. Busby with a version of UNCLE that she used for her Ph.D dissertation[4] on propulsor flow. Her effort involved adaptation and use of that propulsor version for external flow about bodies moving at constant speed. Results for the unappended SUBOFF body and Body 1 were demonstrated.[5]

With the availability of UNCLE.6DOF in early FY97, it became desirable to continue the transition process with this more powerful software. The version of the code for the appended SUBOFF with the addition of sailplanes and a five-bladed propulsor was transferred to CDNSWC along with the grid and sample input files. This version was used at PSU for several maneuvering test cases.[1] The UNCLE portion is quite general and includes many advanced features such as multiblocking for complex geometries, multigrid for convergence acceleration, and dynamic grids which can move with the propulsor or with the vehicle itself. Not included in this initial version are two equation turbulence models, moving appendages, parallelization, and other recent improvements which were not yet ready for further dissemination. The 6DOF portion and coupling is likewise fairly general although some of the code is, of necessity, tailored to the particular application.

A strategy was devised to test UNCLE.6DOF on a series of problems ranging from one as simple as a repeat of the Body 1 calculation of Busby[5] to a full-blown trajectory prediction. Thus the initial calculations involve the UNCLE portion of the code alone and some involve tests that UNCLE has already passed. However, such a thorough process is necessary to verify that the UNCLE routines in UNCLE.6DOF have not suffered from any unintended deleterious modifications. In addition such an effort is necessary for the author and other CDNSWC personnel to gain expertise in the use of this tool.

This report presents the results and experience gained from that portion of this transition and testing process carried out in FY97. Since there appears to be very little available in the way of notes or instructions on the use of any of the versions of UNCLE, annotated input files and other descriptive information are included as well.

TEST CASES

The UNCLE.6DOF code was obtained from The Pennsylvania State University and installed on the SGI Power Challenge system at the Navy Hydrodynamic/Hydroacoustic Technology Center (H/HTC) at CDNSWC. The code, which was being run on IBM equipment at PSU, was found to be quite portable and compiled and executed without much difficulty. Execution errors encountered initially were eliminated by replacing FORTRAN double precision declarations for real variables with a compiler option to specify double precision. CPU usage for the basic UNCLE routines was found to be essentially the same as that reported by Busby[5], 4.78×10^{-4} seconds/grid point/iteration,

for the highest optimizition. A listing of a Makefile used for compilation is provided in Appendix A.

The installed code and input were modified as necessary to carry out the following test cases.

SUBOFF with sailplanes and propeller

This first test case is merely an exercise in running the code with a grid and other input files provided by PSU. The geometry is the SUBOFF configuration with sail, sailplanes, four stern appendages and a five-bladed propeller.[1] The grid consists of over 1.6 million points in 51 blocks. Quantities are nondimensionalized by the body length L and forward speed U. The vehicle is accelerated instantaneously from rest to a constant speed of one and the propeller is abruptly accelerated to a constant rotation rate of 77.2 radians per unit of dimensionless time. The solution is run, as an unsteady problem, to an almost periodic state (unsteady because of the propeller rotation) at dimensionless time of two. (At that time the body has moved two body lengths and most of the starting transients are eliminated). Thus the 6DOF capablities are not invoked. This is accomplished by setting the parameter MOTION = 0 in the input. Such a setting turns off the 6DOF and the code automatically specifies straight ahead motion at constant speed.

In addition to the grid, three other input files are required: UNCLE.INP, which includes most of the basic UNCLE parameters; VEHICLE.INP, which contains vehicle characteristics; and BC.INFO, which includes the UNCLE boundary condition information. Annotated versions of these files are given in Appendices B-D. Array dimensions and multigrid level information must be specified in a FORTRAN include file, PARAMETER.INC, an example of which is given in Appendix E.

For MOTION = 0 the time step (or CFL number) can be input directly. Alternatively, if a negative CFL number is input, a desired number of time steps per propeller rotation can be input, and the code computes the time step accordingly. For MOTION = 1, which invokes the full 6DOF capability, this negative CFL option must be used in the current version of the code. For the test case the negative CFL option was used to start the calculations with 160 time steps per propeller rotation. This resulted in a Δt of about $5\text{x}10^{-4}$. Only one multigrid cycle (with three grid levels) was used per time step and Newton subiterations were not employed (MCYC = 1 and NSUB = 1 in UNCLE.INP). Minimum time stepping (constant time step throughout the field) was used for time accuracy (LTDSG = 0). It was demonstrated[1] that local time stepping (larger time steps in regions with grid spacing larger than the minimum) can be used at the early stages of such a calculation to accelerate its convergence to a periodic state. The calculations proceeded smoothly for about 1500 time steps when results abruptly deteriorated and floating point overflow was encountered. The solution was successfully continued by increasing the number of time steps to 320 per propeller rotation. UNCLE computes forces and moments, pressure and frictional contributions,

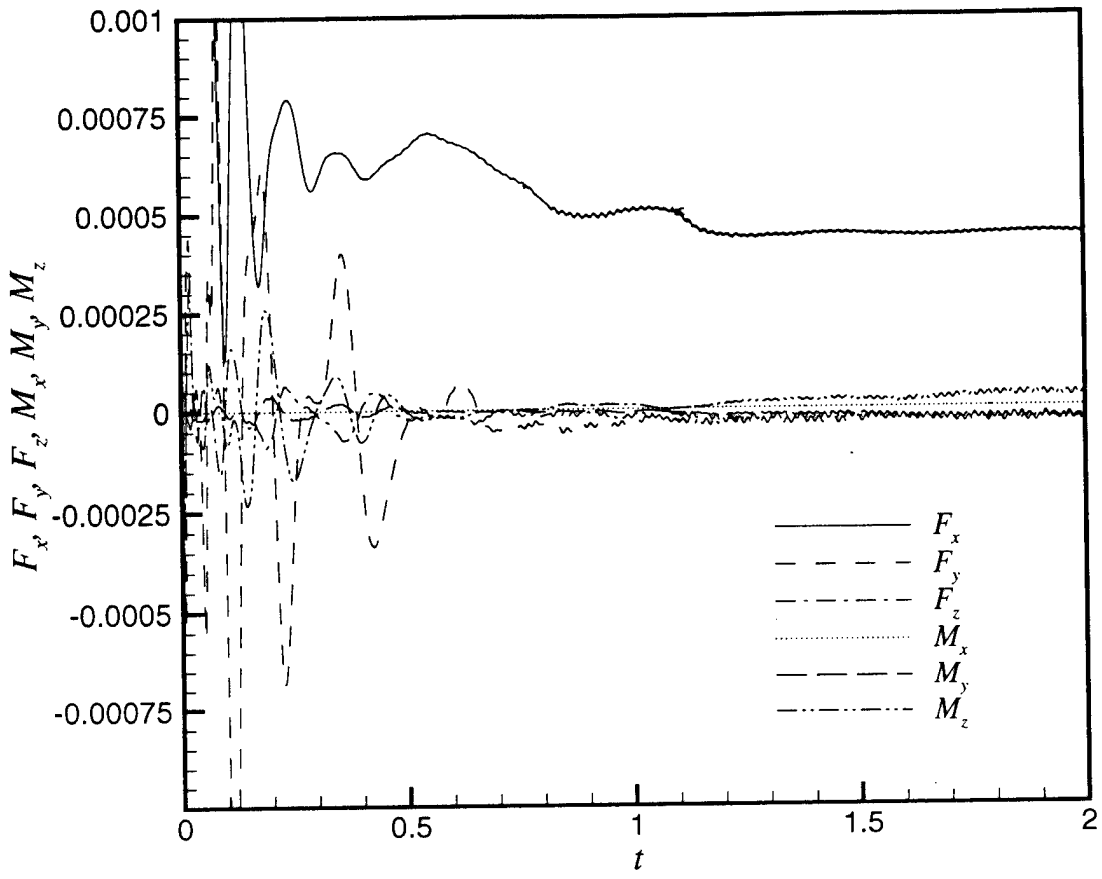


Fig. 1. Computed forces and moments vs. time for modified SUBOFF with propulsor.

and outputs the results to file FORCE.HST. A plot of this file is presented in Figure 1. In UNCLE.6DOF these quantities are computed by integrating the nondimensional pressure and shear stress over all nonslip walls (including the propeller). This amounts to obtaining force and moment coefficients defined by dividing the dimensional forces and moments by $\rho U^2 L^2$ and $\rho U^2 L^3$, respectively. These definitions differ by a factor of 1/2 from the more common force and moment coefficient definitions which are found in some versions of UNCLE. The dominant force is the drag F_x which is seen to settle down, shortly after $t = 1$, to a steady mean value with a low amplitude periodic oscillation due to the turning propeller. This drag agrees with results obtained at PSU[1] and it reflects the fact that the propeller rotation rate is less than that needed for self-propulsion of this body. If the body were not constrained to move at constant speed it would slow down and was shown to do so in unconstrained computations at PSU. The other forces and moments also settle, by $t = 2$, to relatively periodic fluctuations with relatively low mean values. Other output contains the forces and moments on the

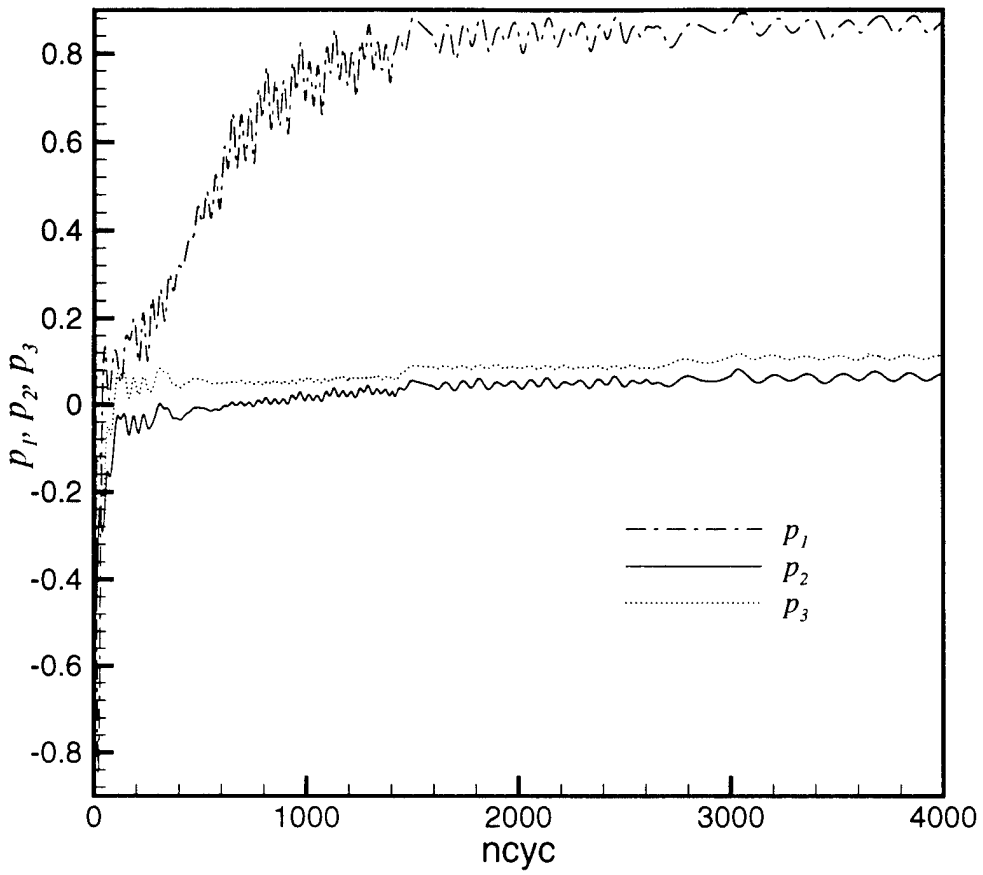


Fig. 2. Computed dimensionless pressures at three locations on the pressure side of a propeller blade vs. time step number.

propulsor alone.

The current version of UNCLE.6DOF outputs several files with time histories of the dimensionless pressure at several selected points on a propeller blade. For example such time histories for three points on the pressure side near mid span of a blade are displayed in Figure 2. The curve labeled p_1 is taken from a centroid near the leading edge while p_2 is near mid chord and p_3 is near the trailing edge. As with the force histories, these plots show a disappearance of starting transients and an approach to periodic flow as the blade cuts through the appendage wakes. The apparent changes in frequency are the result of changes in Δt at about ncyc = 1500 and ncyc = 2600. Also about 100 time steps of data were lost around ncyc = 1500. These outputs of pressures are hard coded for this geometry with block numbers and grid point index values. Thus a user wishing to obtain such output for a different geometry will need to make appropriate changes in the code or write a postprocessor to extract the desired information from the full solution file, UNCLE.DMP.

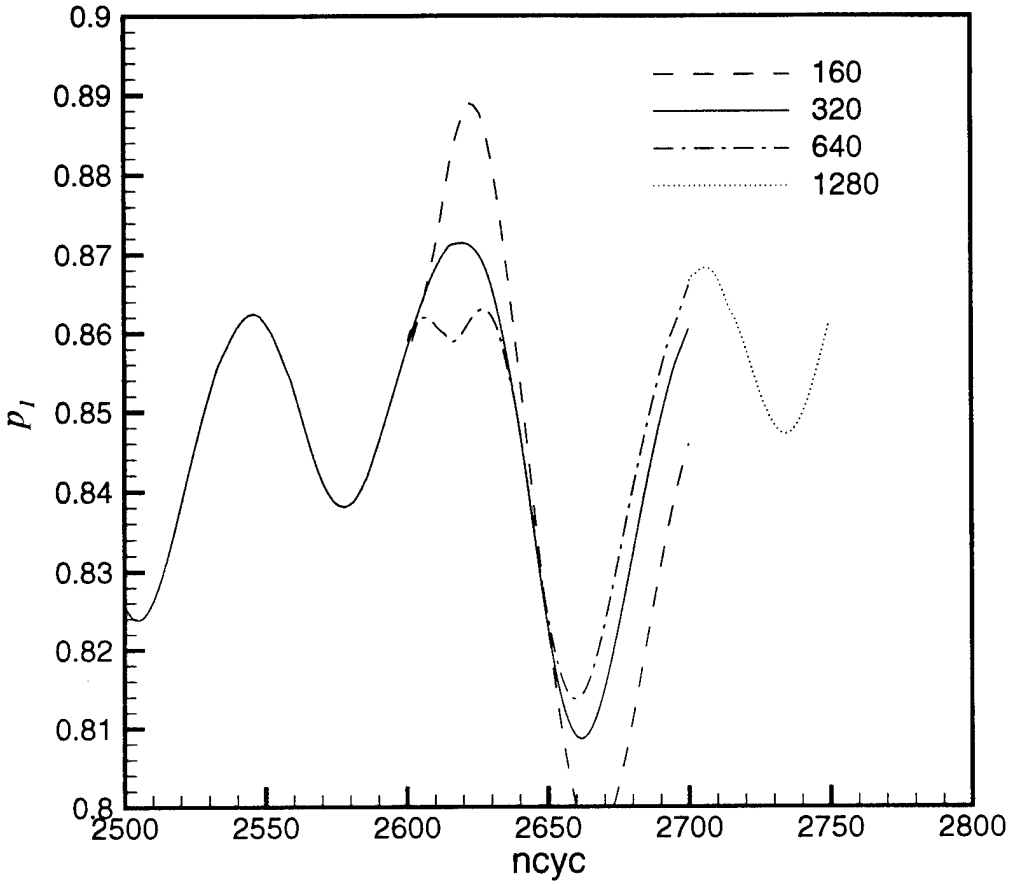


Fig. 3. Pressure history with various numbers of time steps per propeller revolution.

A crude experiment on time accuracy was carried out at ncyc = 2600 and the results are displayed in Figure 3. The calculations were proceeding smoothly with 320 time steps per revolution. The interval from ncyc = 2600 to 2700 was repeated with two other step sizes. A doubling of the step to 160 per revolution resulted in a noticable increase in the amplitude of the fluctuations in the pressure history near the leading edge of the blade. On the other hand, a halving of the step size to 640 per revolution resulted in a reduction of the amplitude but the amount of change due to the halving was not as large as that due to the doubling. The halving introduced a local minimum at about ncyc = 2615. To assure that this was not an anomaly introduced by the change in step size, the step was further halved to 1280 per revolution at ncyc = 2700. The time history is computed smoothly through this latter change. These results indicate that the results are convergent with respect to time step size but do not prove time accuracy. The computations are continued beyond this time with 640 steps per propeller revolution.

Figure 4 shows a color contour plot of the pressure on the body surfaces. This picture was made using the FAST visualization software running on an SGI workstation at the H/HTC with the PLOT3D format output file from UNCLE.6DOF.

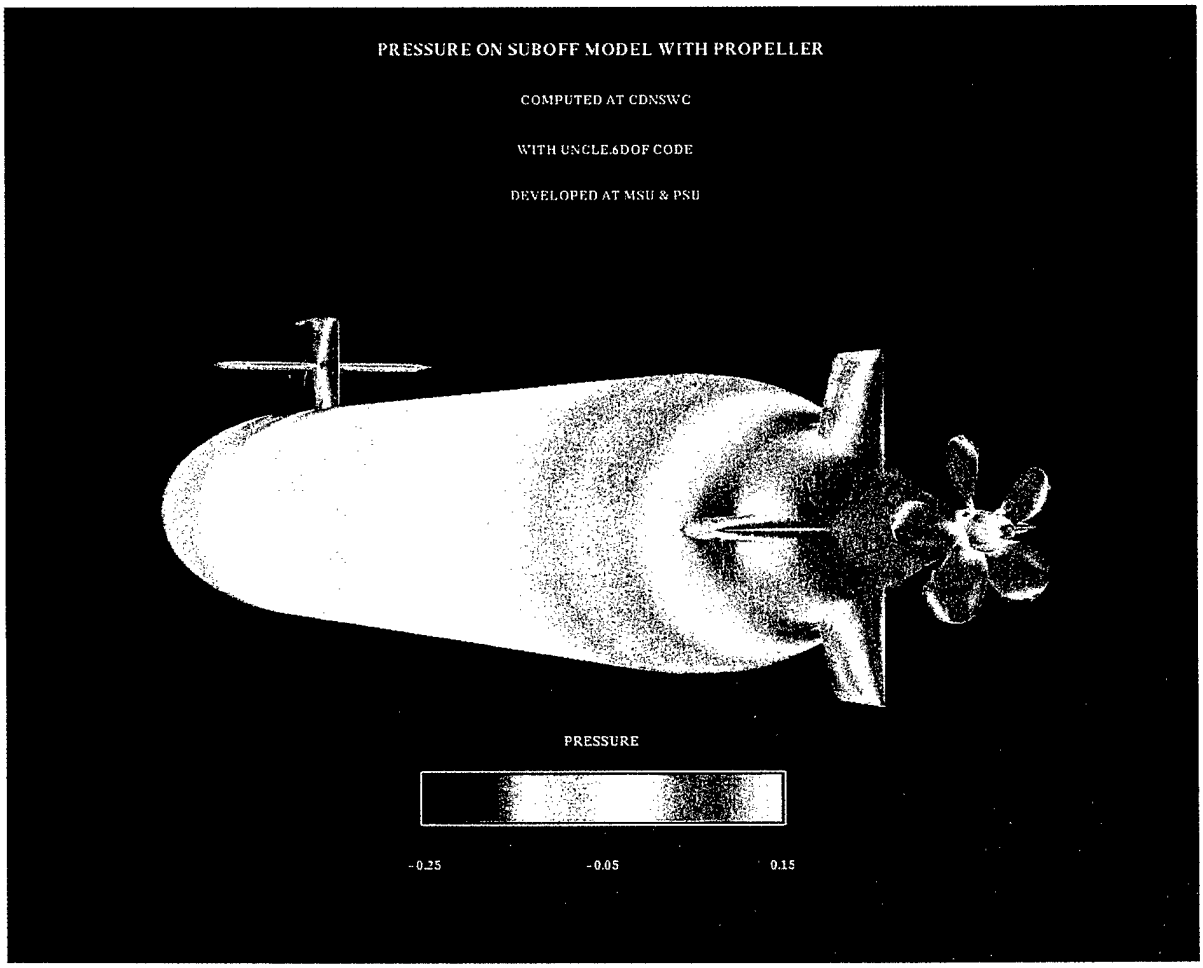


Fig. 4. Computed pressure on the surface of SUBOFF with propeller.

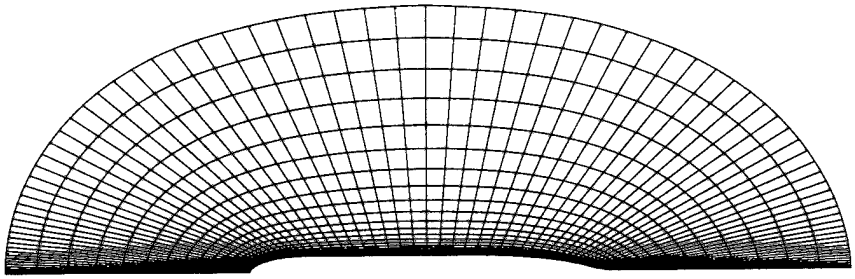


Fig. 5. Body 1 profile and grid surface.

Body 1 at zero degree incidence

The previous discussion demonstrated that the transitioned UNCLE.6DOF code produced essentially the same results as those obtained at PSU with grid and other input files obtained from PSU. Most of the main features of the UNCLE portion of the code, such as multiblocking, time accuracy and a rotating propeller were included. As a next step it was desirable to test the code on a much simpler problem but with locally generated grid and input files. For this purpose DTMB Body 1, an unappended axisymmetric body was chosen. Use of Body 1 allows comparison with the results of model tests and previous computations including the UNCLE computations of Busby[5].

The profile of Body 1 and a representative grid surface are displayed in Figure 5. The grid is the same 81 x 41 x 17 90° wedge O-grid used by Busby. It was generated algebraicly at CDNSWC. A finer 129 x 81 x 33 grid was also used and it was verified that both grids provided essentially the same results. Initially the case of straight ahead motion at 0° angle of attack is considered. The Reynolds number, based on body length, is $Re = 6.6 \times 10^6$. In general, this parameter is specified through input of the water density and viscosity as well as reference length and velocity scales in VEHICLE.INP. It was found that for the current problem it is sufficient to set the density equal to the desired Re and to set these other parameters equal to one. The code can be applied, through the input, in two ways, with either the body moving and the farfield flow at rest (IDYN = 1 in UNCLE.INP and VINF = 0.0 in VEHICLE.INP) or with the body at rest in a nonzero farfield flow (IDYN = 0 and VINF = 1.0). It was found that precisely the same forces are obtained from the two cases. The body (or flow) is accelerated abruptly from rest to dimensionless speed one and the steady state approach (ISTATE = 1 in UNCLE.INP) is taken with local time stepping (LTSDG = 1 in UNCLE.INP) to accelerate convergence.

Fortunately, few code changes were found necessary for running the current case. In the original code the grid coordinates were rescaled by the body length after being read in subroutine MAIN. This coding was geometry dependent and needed to be changed. The axis boundary conditions at the nose and tail of the body, in subroutine BCI, are

block structure dependent and needed to be recoded for the current one-block 90° wedge configuration. Other boundary conditions - nonslip, reflective symmetry, interblock, and farfield - are specified through the BC.INFO input file. Although the UNCLE.6DOF code was written assuming the presence of a rotating propeller, it was found that for such a simple problem, with straight ahead constant speed motion (MOTION = 0), that the propeller portions of the code do not interfere with the solution process. The propeller input, which is still read in, was left unchanged from the previous (SUBOFF) case. Since there is only one block in this case the code never gets to the propeller blocks specified in the input. However, it might be necessary in some cases, to adjust the propeller input or suitably rewrite the code to avoid unwanted interference from the propeller portions of the code.

The code was run for 500 iterations with CFL = 5. Only one multigrid cycle was employed per time step with three multigrid levels. Newton subiterations were not used. The drag coefficient converged to a value of about 8 x 10^{-4} by the completion of 200 iterations. This compares favorably with a measured value of about 7.7 x 10^{-4} for $Re = 10 \times 10^6$ *. The cpu time reported by Busby of about 4.78 x 10^{-4} seconds/grid point/time step on an H/HTC SGI Power Challenge R8000 processor was confirmed.

The pressure coefficient and skin friction on the body surface, computed with the fine grid are compared with measured data[6] in Figures 6 and 7. The computed results are essentially the same as those of Busby.

Body 1 at angles of attack

It is one thing to get the pressure and skin friction correct on an axisymmetric body at zero degree incidence and quite another to predict correct forces at angles of attack. Body 1 has been run at angles of attack up to 20° for $Re = 12 \times 10^6$. The fine grid in a port side 129 x 81 x 65 half-field version was used. The force coefficients computed by the code are presented in Figures 8-10 where they are compared with the computed values of Sung et al. which were shown to agree well with measured data.[7] It is well-known[8] that use of the standard Baldwin-Lomax turbulence model yields a lift force which is much lower than reality and a pitching moment higher than reality for bodies at high angles of attack in this Reynolds number range. This is borne out by the two points, represented by squares, for $\alpha = 20°$ plotted in Figures 8 and 9.

The approach typically used with UNCLE to modify the Baldwin-Lomax model[1] has avoided the general coding of the approach suggested by Degani and Schiff[8]. Instead two constants have been employed: (1) a search parameter and (2) a boundary layer thickness parameter. The search parameter is a value of the F function beyond which the search for F_{max} is terminated. The thickness parameter is a value of the boundary layer thickness divided by the local body radius above which the search for F_{max} stops once the first local maximum in F is found.

*R. Curphey, private communication.

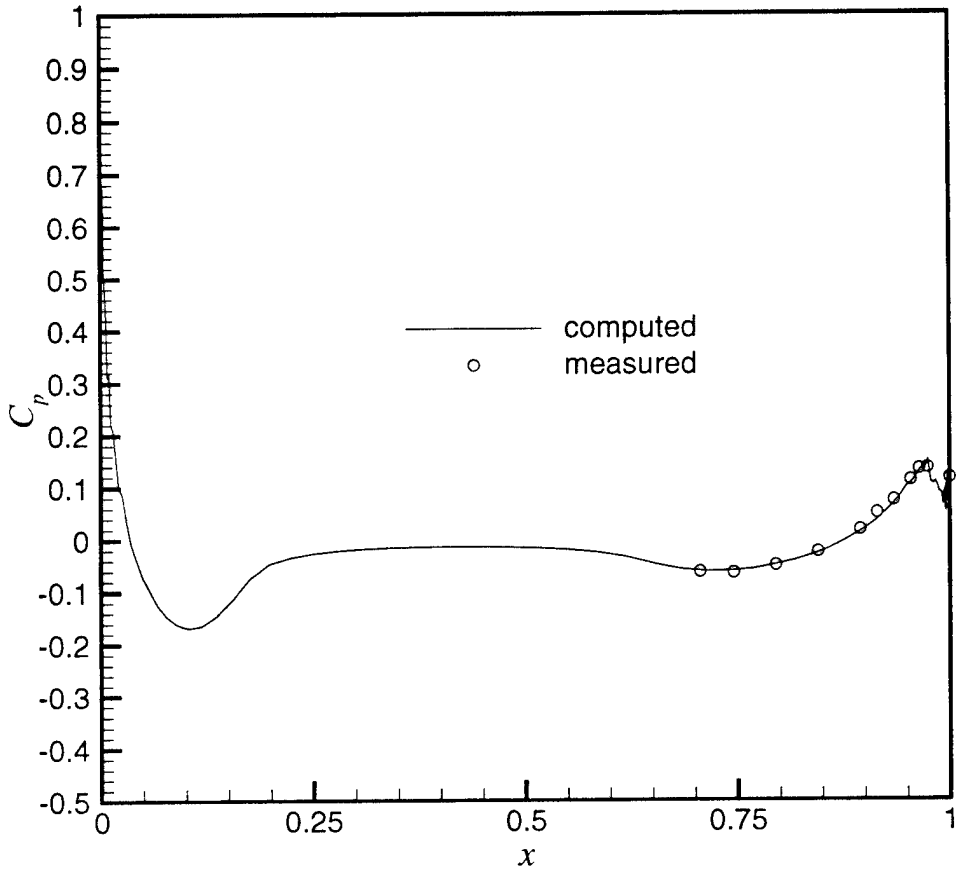


Fig. 6. Comparison of computed and measured pressure coefficient on Body 1.

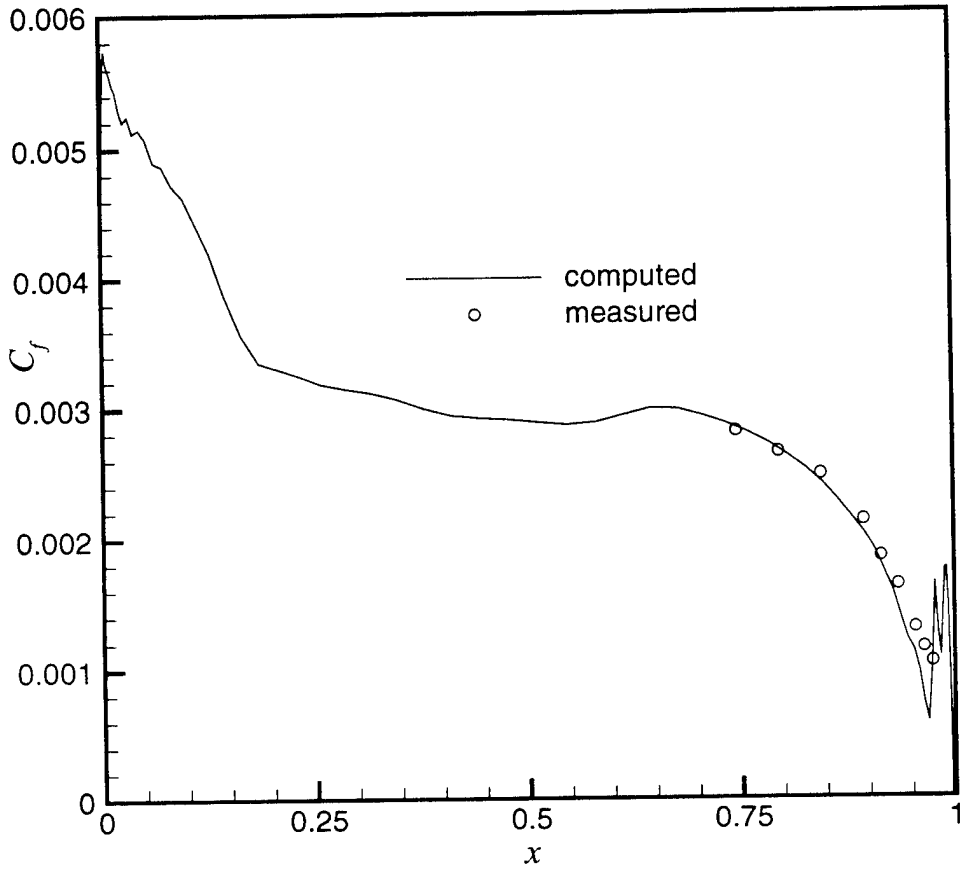


Fig. 7. Comparison of computed and measured skin friction coefficient on Body 1.

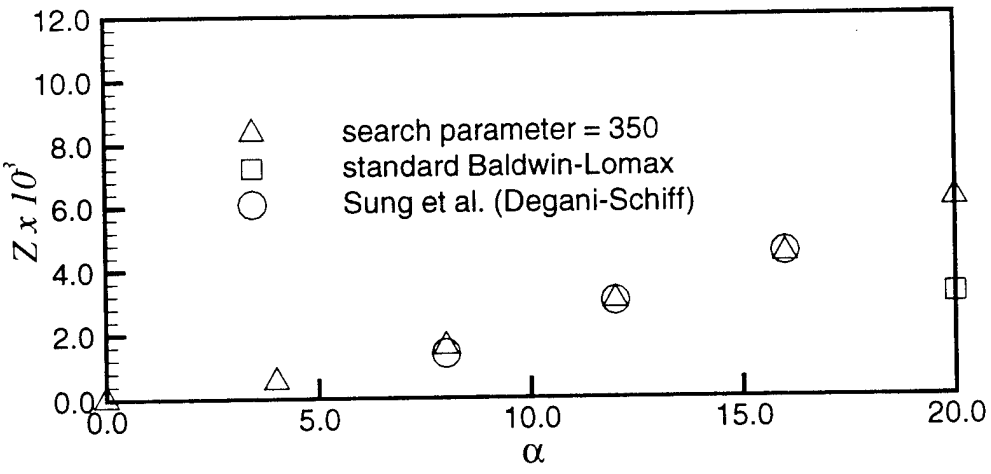


Fig. 8. Computed lift coefficient for Body 1 at various angles of attack.

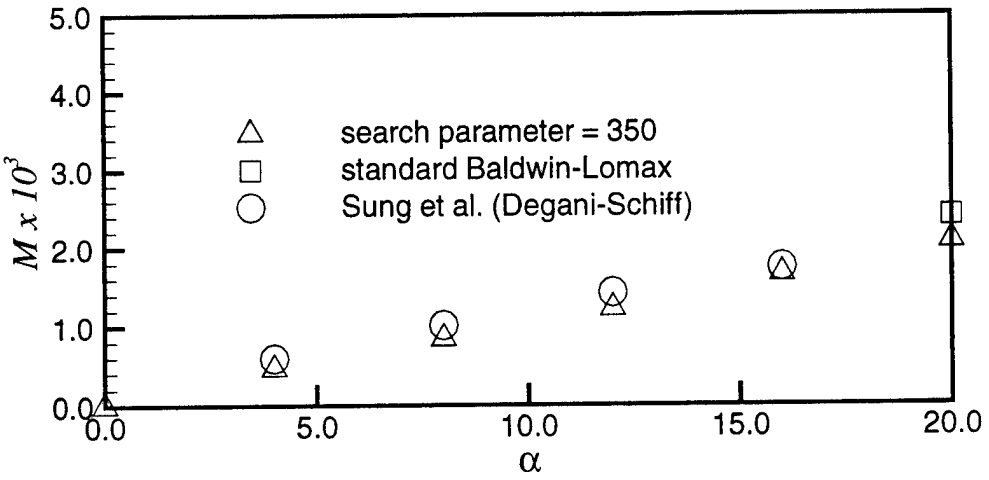


Fig. 9. Computed pitching moment coefficient for Body 1 at various angles of attack.

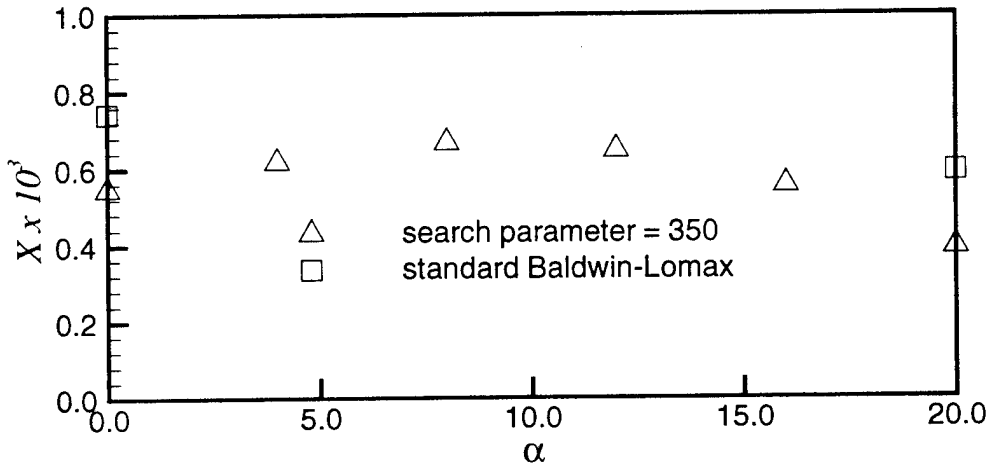


Fig. 10. Computed drag coefficient for Body 1 at various angles of attack.

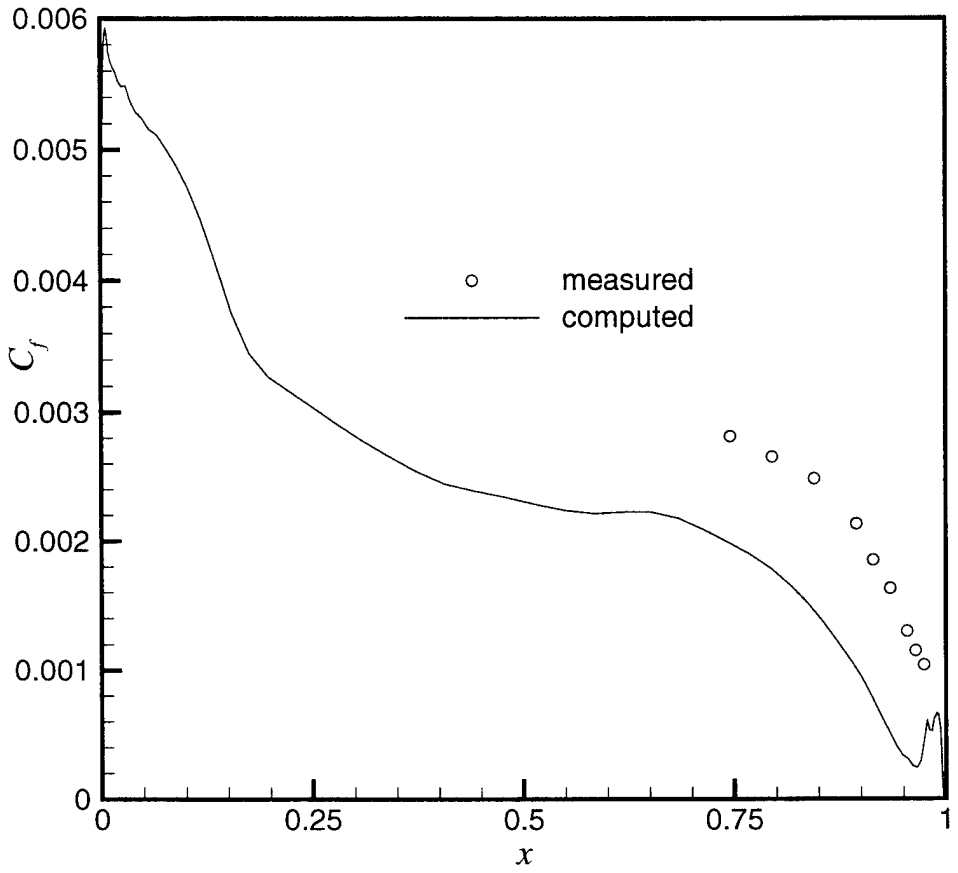


Fig. 11. Comparison of measured skin friction coefficient with values computed using a search parameter of 350.

The version of the code originally obtained had only a standard Baldwin-Lomax subroutine. Other versions of the Baldwin-Lomax subroutines were obtained from MSU in which only the search parameter was employed. The points so labeled in Figures 8 and 9 were obtained with these modified routines with a search parameter of 350. This search parameter contrasts with the value of 750 reported for the unappended SUBOFF body.[1] The value 350, which is used here for all angles of attack, was chosen, by trial and error, to obtain points which compare favorably with measured data and with previous computations.[7] However, Figure 10 shows that the drag values predicted with the standard Baldwin-Lomax model appear to agree better with the measured data than those predicted using the search parameter. Such an underprediction of drag, which corresponds to an underprediction of skin friction (Figure 11), is similar to results previously presented for the unappended SUBOFF body at angles of attack.[1] A

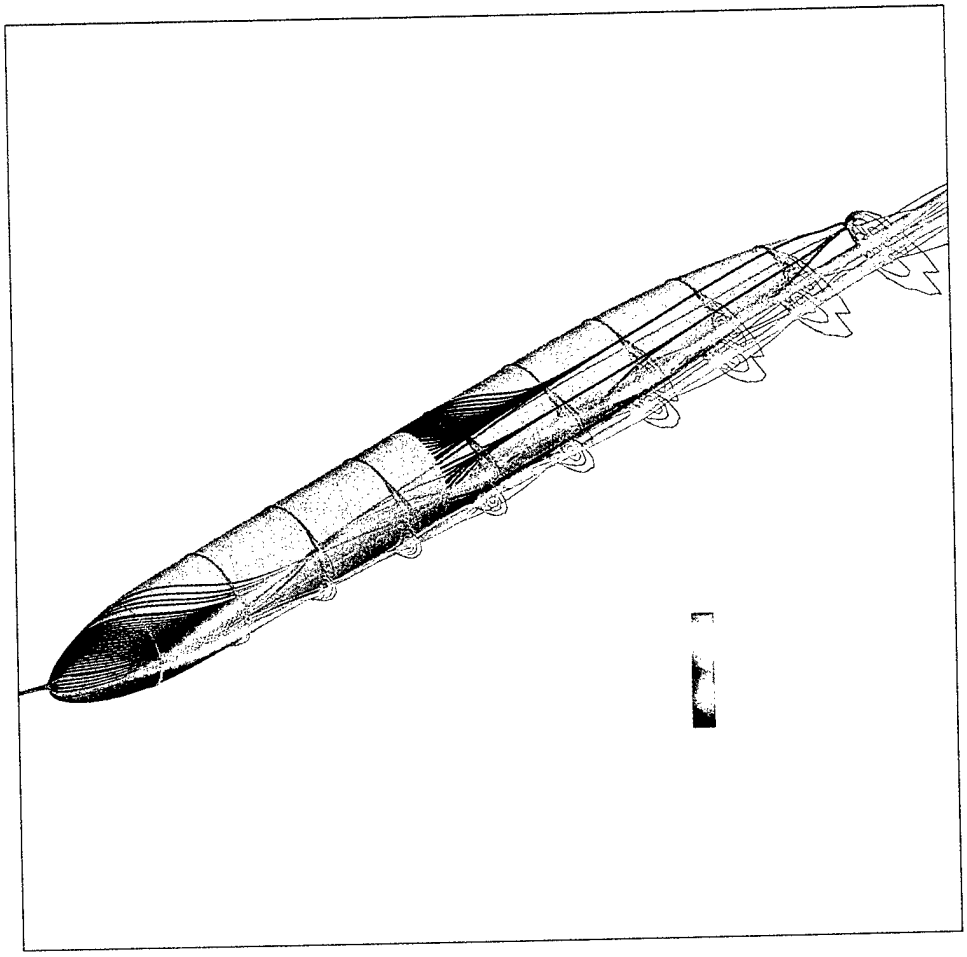


Fig. 12. Computed streamwise vorticity and streamlines for Body 1 at 20° angle of attack.

partial remedy is to determine optimal search parameters at every angle of attack of interest. Also, previous experience indicates that drag was very sensitive to resolution in the axial direction.[†]

A visualization of the flow for $\alpha = 20°$ is shown in Figure 12. This picture and the following color visualizations were created using the Fieldview (©Intelligent Light) visualization software. The half-field results were reflected to obtain the full-field picture. Near-surface streamlines, colored by streamwise vorticity magnitude, are started at a rake near the bow. These coalesce along separation lines and lift off the surface thereafter passing near the cores of two vortices in the lee of the body. These vortices are visualized by lines of constant streamwise vorticity in planes intersecting the body at numerous stations. Note that the version of Fieldview employed did not

[†]L. Taylor, private communication.

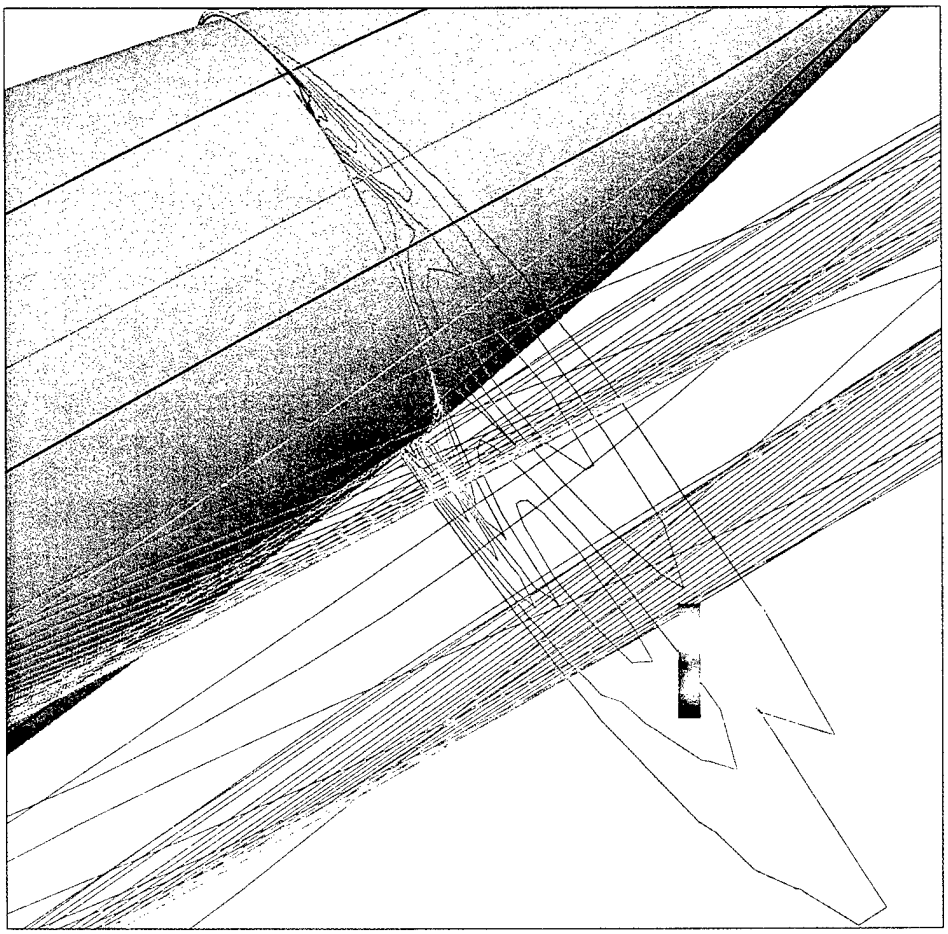


Fig. 13. Closeup of computed streamwise vorticity and streamlines for Body 1 at 20° angle of attack.

change the sign of the vorticity upon reflection. Another set of streamlines, confined to remain near the body, originates from a rake near midbody. These coalesce into primary and secondary separation lines. A close-up of part of the field near the stern is shown in Figure 13. Some of the details of both the primary and secondary vorticity can be seen.

Body 1 in a steady turn

The previous section demonstrated a force computation capability whose accuracy is subject to realistic turbulence modeling, as is the case for all RANS codes. While ultimate predictive accuracy depends on improved turbulence modeling, it is still possible to learn much from the current code and worthwhile to continue to test its many features. To begin to test its capabilities on a more complex motion, a steady turn is chosen. This is a circular trajectory and is called steady since the flow, in a reference

frame relative to the body, approaches a steady state if the turn continues until starting transients have been eliminated. Since UNCLE.6DOF uses a fixed reference frame, the flow field velocities will not be steady and the problem must be run as unsteady. Ordinarily, one would use the relative frame to minimize the work in computing such a flow field. However this problem provides a good test of the general trajectory and time dependent capabilities of UNCLE.6DOF.

It was desired to first test the basic capability of UNCLE to compute the forces on a turning body. To do this relatively quickly, a so-called "rotating arm" version of UNCLE was obtained from MSU. This is a version set up specifically to simulate a test of a body attached to the rotating arm at CDNSWC and moved in a steady turn. The code solves the equations in the relative frame as a steady problem with local time stepping to accelerate the convergence. This code was received with a SUBOFF grid and input but required only minor modification to apply it to the same Body 1 grid described in the previous section. Because the flow about the body in a turn (with zero angle of pitch) exhibits symmetry about a horizontal plane, the port side half-field grid used for the body at an angle of attack must be replaced by either a top or bottom half-field grid. The body was considered to be effectively fixed to the rotating arm at a point 0.4530 body lengths from the bow. This number is not included in the input and needed to be changed in the code. A dimensionless turning rate, also appearing explicitly in the code, is defined by $r' = L/r$ where r is the dimensional distance corresponding to the rotating arm length. The speed of the body in the turn is specified by the input Reynolds number of $Re = 12 \times 10^6$. Computations were carried out with CFL = 1 and the computed force and moment coefficients converged to the values indicated by triangles in Figures 14-16 within a few hundred iterations. These are the forces and moment that would be measured by the rotating arm guages and the sideforce is accordingly the sum of the hydrodynamic side force predicted by the code plus the centrifugal force applied by the arm to keep the body moving on the desired circular path. The results were obtained with the Baldwin-Lomax search parameter of 350 as used for the body at an angle of attack. The bounds of the measured data are those published by Sung et al. The computed values of the side force shown in Figure 14 fall roughly in the middle of the range of data. They agree well with those of Sung et al. for the lower turning rates. At the higher rates the results of Sung et al. are near the bottom of the range of data. While the computed values of yawing moment presented by Sung et al. seem to be somewhat low compared with the data (Figure 15), those computed by UNCLE seem to be high by a similar amount. As for the body at an angle of attack, the computed drag values (Figure 16) are somewhat low. (Sung et al. did not publish results for drag.)

The forces computed by the rotating arm code are subject to the usual array of errors including spatial discretization. The results for the same problem with UNCLE.6DOF will depend, in addition, on time accuracy. While the forces should approach a steady state, the computation of those forces, even at their steady state, will involve time

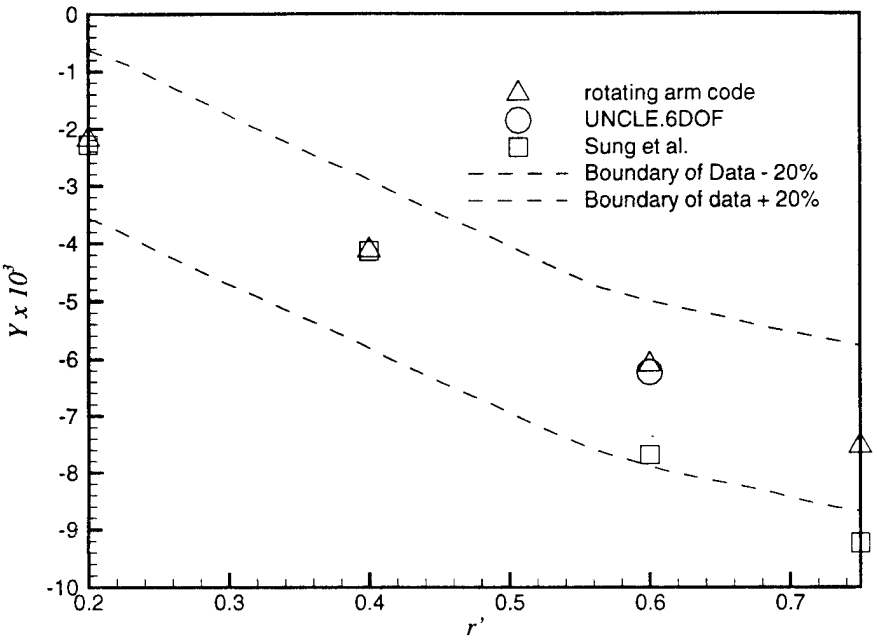


Fig. 14. Computed sideforce coefficient for Body 1 in steady turns at various turning rates.

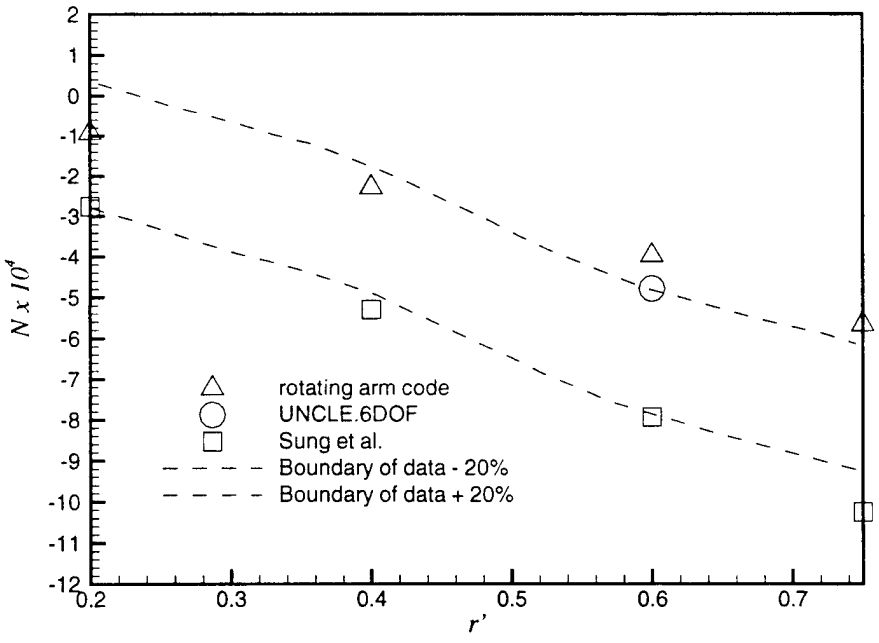


Fig. 15. Computed yawing moment coefficient for Body 1 in steady turns at various turning rates.

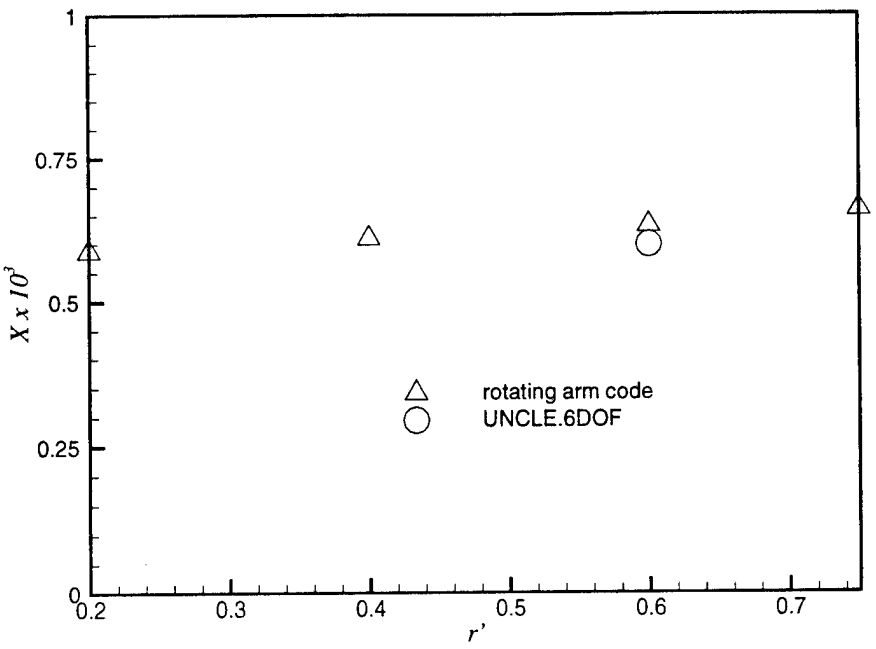


Fig. 16. Computed drag coefficient for Body 1 in steady turns at various turning rates.

derivatives of the velocities which must be computed accurately to achieve good force values. Thus the current problem is a test of the time accuracy of UNCLE.6DOF. As mentioned previously, the code has options - MOTION = 0 to invoke straight ahead motion at constant speed, and MOTION = 1 to compute a trajectory using the full six degrees of freedom capability. In addition there is a subroutine PRESCRIBE included to allow the user to easily include code to specify an arbitrary trajectory. A few difficulties were encountered in trying to use this subroutine with MOTION = 1, as apparently intended, to prescribe a steady turn. In trying to solve these difficulties the author created a new option, MOTION = 2, to use for a prescribed trajectory. When this option is invoked, the call to the subroutine (NEWGRIDU) that specifies straight ahead motion at constant speed is replaced by calls to subroutines MOVE and NEWGRID which handle the grid movement for more general trajectories. In subroutine MOVE the call to subroutine SIXDOF is replaced by a call to PRESCRIBE which consists of just a few lines of code specifying a steady turn. The remainder of the code is the same as for MOTION = 0.

Three reference point coordinates are defined in subroutine MOVE to specify a reference plane for use in the grid movement. These coordinates were defined for the propelled SUBOFF geometry as values at grid points referred to in the code by their block number and position indices in that grid. Thus, this section had to be recoded for the Body 1 grid. For the SUBOFF grid, the first point was at the bow, the second at the

stern, and the third on the centerplane near the sail. Initially, for Body 1 a top half-grid was used and the code was changed to specify three points at the bow the stern and the centerplane near midbody. This did not result in a correct grid movement. However, it was found that when using a bottom half-grid the grid movement is correct. This might be connected with an assumption of UNCLE.6DOF that the coordinate system be such that the sail points in the negative vertical coordinate direction.[‡] The specification of the three reference points as well as limitations on the coordinate system need to be further clarified.

Recoding was also needed in the Baldwin-Lomax model subroutines received from MSU. Code designed to implement a wake model downstream of the stern of a body moving in the negative x-direction was found to implement such a model along the body itself whenever the body was facing in the positive x-direction. This code was eliminated. Such recodings are typical with developing RANS software and are among the features that make use of such software difficult and even risky for casual users.

The computation of the steady turn with UNCLE.6DOF was started from the converged result for straight ahead motion. This straight ahead starting condition was computed with MOTION = 2 but as a steady computation with CFL = 10.0 and local time stepping. At $t = 2500.0$[§] the computation was restarted with the turning added to the forward speed and as an unsteady computation with a time step of 0.01 everywhere. Thus, the body moves through about 1/3 of a degree in one time step. Initially it was run with two multigrid cycles per time step and a halving of the time step showed that the results were independent of Δt. However, the studies presented in the next section indicate that convergence of the results in Δt is not sufficient for time accuracy. The number of multigrid cycles can effect time accuracy independently of the time step size. Therefore the steady turn was rerun with more attention paid to the number of multigrid cycles.

The time histories of the force and moment coefficients computed by UNCLE.6DOF for the steady turn with $r' = 0.6$ are shown in Figure 17. The side force coefficient C_y represents the hydrodynamic force without the addition of the inertial force applied by the rotating arm. The converged result for straight ahead motion, up to $t = 2500.0$, shows essentially zero side force and yawing moment. The turn begins at that time and immediate large pressure pulses, due to the impulsive start of the turn, are followed by transients which mostly disappear by $t = 2502.0$ or by the time the body has moved through two body lengths or about 70°. At $t = 2502.0$ the number of multigrid cycles per time step was increased from two to four. The force values change with the most noticable change being a drop in the side force. These changes indicate that the solution obtained with only two multigrid cycles per time step is not time accurate enough to provide accurate force values. The number of multigrid cylces was further increased to

[‡]F. Davoudzadeh, private communication.

[§]For steady problems, the code defines "time" as the product of the CFL number and the number of iteration cycles.

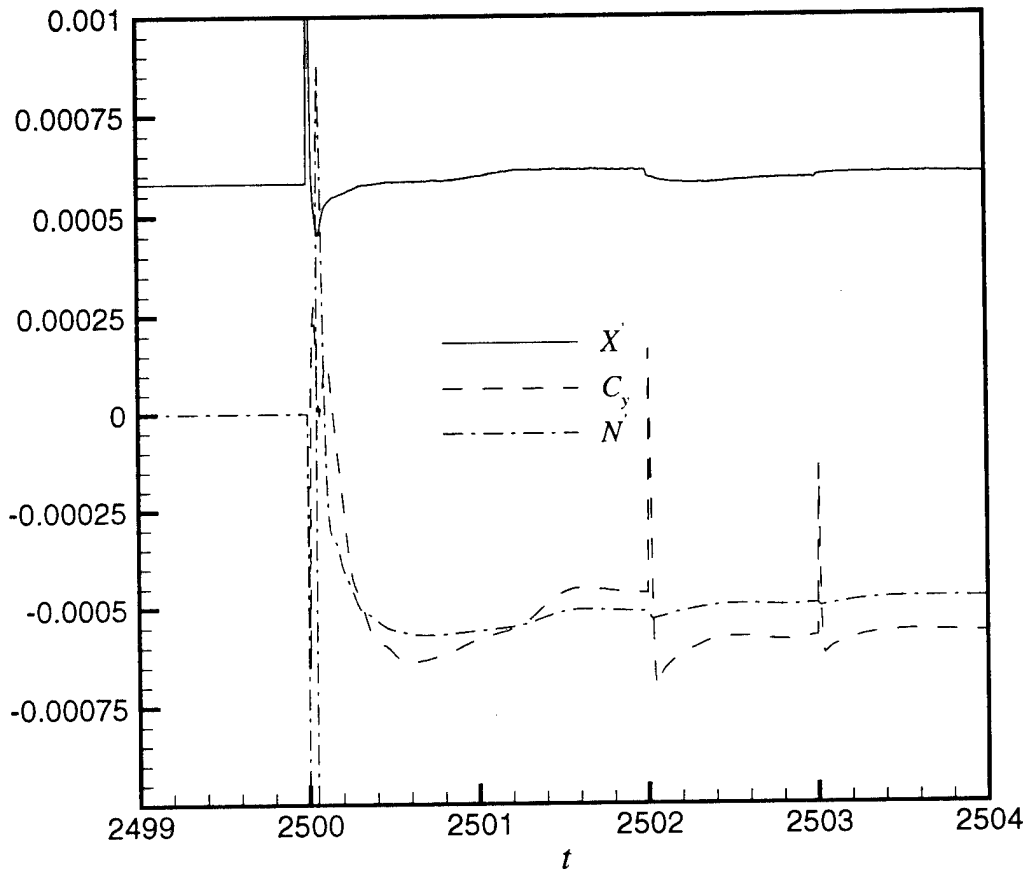


Fig. 17. UNCLE.6DOF computed force and moment history for Body 1 in straight flight abruptly changed to a steady turn with $r' = 0.6$.

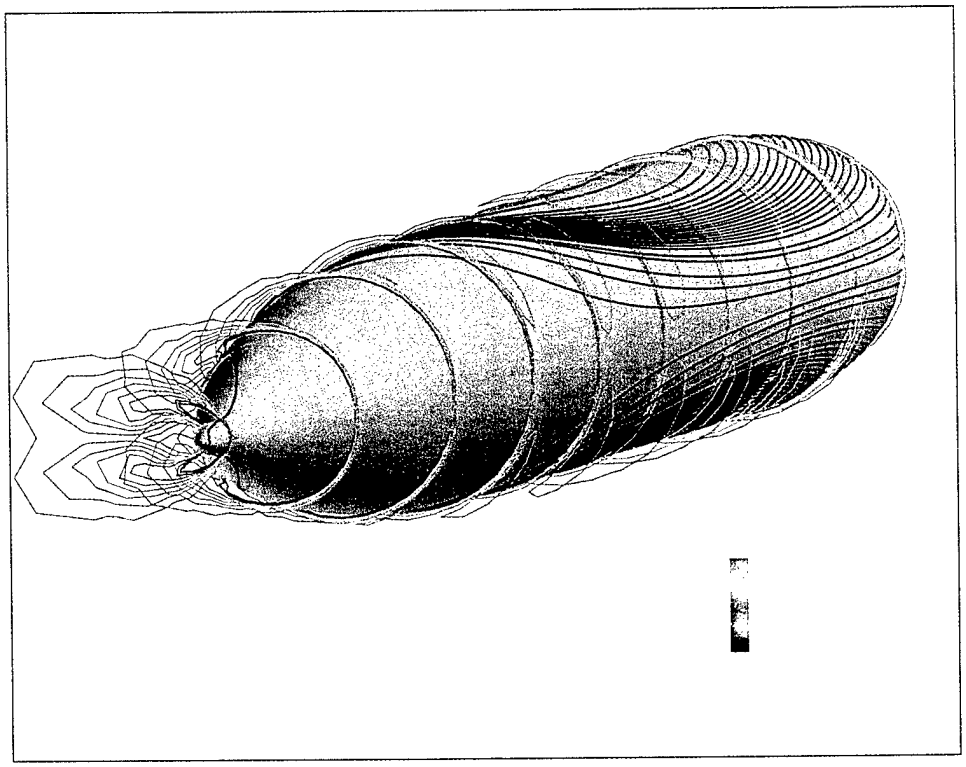


Fig. 18. Computed streamwise vorticity and streamlines for Body 1 in a steady turn ($r' = 0.6$).

eight per time step at $t = 2503.0$. The effect of this increase is smaller than that of the previous increase indicating convergence in multigrid cycles and movement toward time accuracy. The almost steady force values obtained at $t = 2504.0$ with eight multigrid cycles per time step, with the side force coefficient having the inertial force added, are plotted in Figures 14-16 for comparison with the values computed by the rotating arm code. The agreement is a positive result for the arbitrary trajectory and time accuracy capabilities of UNCLE.6DOF. The remaining differences are presumably due partly to small remaining time inaccuracy in the UNCLE.6DOF results and partly to the fact that the rotating arm code contained a Baldwin-Lomax wake model which, because of its lack of generality, was turned off in the UNCLE.6DOF computation.

A visualization of the flow with on body streamlines and streamwise vorticity contours is shown in Figure 18. The reversal of the crossflow direction is apparent.

Body 1 accelerating from rest

The ability to predict accurately a trajectory for a maneuvering body depends heavily on the capacity to compute the instantaneous forces with sufficient accuracy throughout the maneuver. This, in turn, is dependent on the ability to determine the flow field

to a high degree of accuracy. Numerical errors can arise through both spatial and temporal discretization. While much is known about strengths and weaknesses of RANS codes as applied to the solution of steady problems, considerably less is known about their performance in unsteady situations. Therefore, there is much work to be done to expose the effects of temporal discretization and validate RANS codes such as UNCLE.6DOF for unsteady problems such as maneuvering.

Some of the above results give indication of time accuracy and of convergence with respect to time step size. However, further effort is needed to verify time accuracy. It would be ideal to have an analytic solution for comparison, but such solutions are virtually nonexistent, at least for viscous flow. UNCLE was previously applied to the challenging flapping foil unsteady problem.[9] Comparison of results with the measured data were promising, but inconclusive because of the numerous possible sources of computational and experimental error. Application of UNCLE to accelerating circular cylinders was also encouraging.[10] Computed pressure coefficient results on a circular cylinder accelerating in an inviscid fluid showed excellent agreement with an analytic solution. Velocity profiles and wake growth agreed well with measured data for a cylinder abruptly accelerated in a viscous fluid. The basic UNCLE solver was also applied to plunging, turning and pitching maneuvers of a prolate spheroid.[1] Comparison of computed body pressures and separation locations with measured data were favorable but with some discrepencies, particularly at higher angles of attack.

To provide further experience and begin to generate some information on the computation of forces in unsteady situations, UNCLE.6DOF has been applied to Body 1 undergoing a prescribed acceleration from rest to a constant speed. An abrupt acceleration (discontinuous velocity) requires a vorticity sheet at the body surface and infinite force. A discontinuous acceleration exhibits a spatial discontinuity of vorticity gradient and a temporal discontinuity in force. Both types of acceleration require special numerical treatment near startup for accurate force computation.[11] To avoid such difficulties a continuous acceleration is considered:

$$dU/dt = \begin{cases} 0 & \text{for } t < 0 \\ 400t & \text{for } 0 \le t \le 0.05 \\ -400(t-0.1) & \text{for } 0.05 \le t \le 0.1 \\ 0 & \text{for } 0.1 < t \end{cases} \tag{1}$$

where U is the speed of the body, t is time, and quantities are made dimensionless with respect to body length and speed at the end of the acceleration period. It can be seen that the body accelerates from rest at $t = 0$ to $U = 1$ at $t = 0.1$. This is an unrealistically rapid acceleration but serves to provide a severe test to the numerical scheme. The acceleration was implemented by modifying the code in subroutine NEWGRIDU, which normally enforces constant speed, and running with MOTION = 0. It was also verified that running with MOTION = 2 and inserting the proper code into subroutine PRESCRIBE provided identical results. The solution was run as turbulent even though

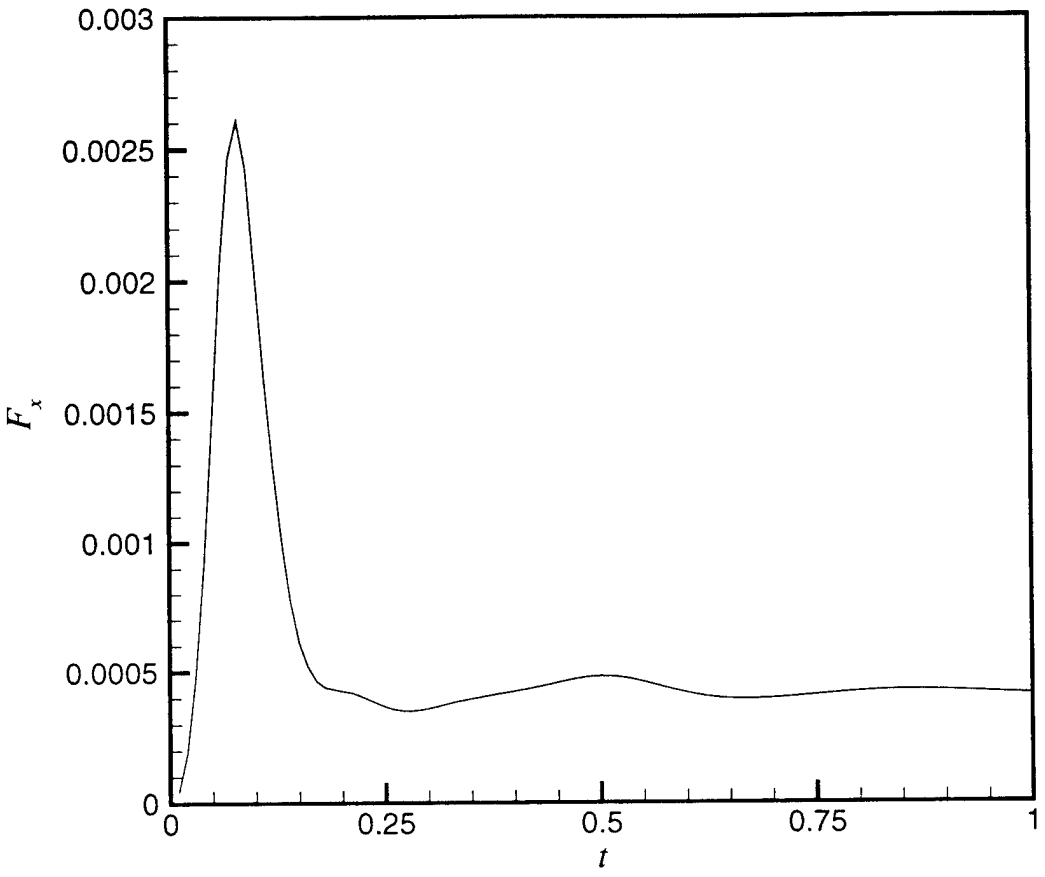


Fig. 19. Computed axial force for Body 1 accelerated from rest to a constant speed with $\Delta t = 0.01$ and 1 multigrid cycle.

started from rest. As will be seen, the main result of interest is the inviscid pressure force so turbulence issues are not critical. To facilitate the running of numerous cases, the coarse grid is used in a port side 81 x 41 x 33 half-field version.

The axial force computed with a time step of 0.01 and with one multigrid cycle per time step is displayed in Figure 19. The force history appears reasonable with a high peak during the acceleration period and then a lower constant value with some low level transients after the acceleration. However, with only ten time steps covering the acceleration period one might suspect that it is not well resolved. The drag during the acceleration period should be dominated by the pressure component which, in turn, would be expected to be dominated by the inviscid acceleration reaction. This component should be proportional to the acceleration, rising linearly from zero at $t = 0.0$ to a peak at $t = 0.05$ and dropping to zero again at $t = 0.1$. The peak actually occurs closer to $t = 0.1$ than $t = 0.05$ and this might be suspected to be inaccuracy resulting from such a large time step.

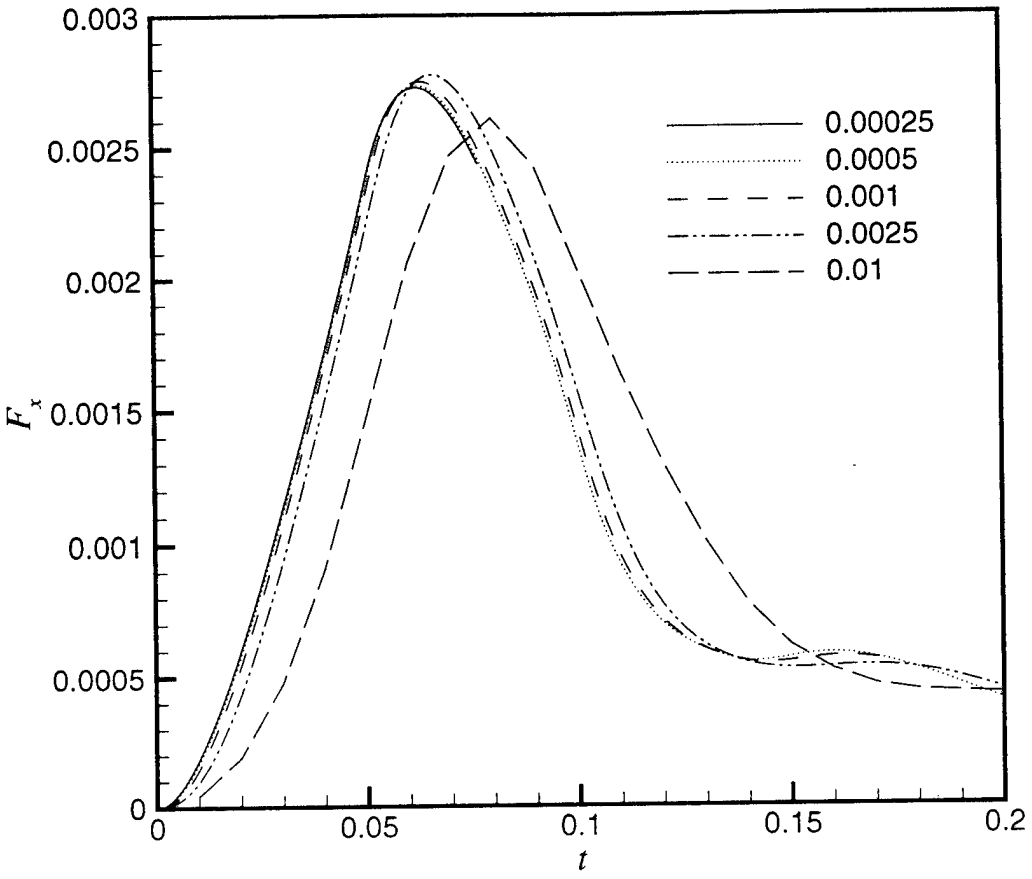


Fig. 20. Computed axial force for Body 1 accelerated from rest to a constant speed with various time step sizes and 1 multigrid cycle.

Figure 20 shows time histories of axial force computed with various values of the time step. As Δt is reduced convergence occurs with the results for $\Delta t = 0.00025$ being almost identical to those for $\Delta t = 0.0005$. However, the initial rise is far from linear and the peak remains noticably later than $t = 0.05$. Apparently there is convergence to a solution which is not time accurate.

The basic UNCLE approach is iterative. Newton's method forms the heart of the scheme for solving the nonlinear system of equations resulting from the time discretization of both the steady and unsteady formulations. Newton iterations should assure time accuracy of unsteady solutions. The number of Newton iterations per time step is specified by the variable NSUB in file UNCLE.INP. Each Newton iteration involves the solution of a linear system of equations which is itself solved iteratively using Gauss-Seidel iterations. The number of Gauss-Seidel iterations per Newton iteration is determined by the parameter ISGS in UNCLE.INP. When multigrid is employed, the Newton iterations are carried out at each multigrid level. The number of multigrid

levels is specified in the PARAMETER.INC FORTRAN file by the variable MG. The number of multigrid cycles, that is the number of times that the code cycles through the various multigrid levels, is determined by the variable MCYC in UNCLE.INP. Thus, the code has several nested iterative levels within a time step. The outermost iteration is the multigrid cycle. This is followed by the multigrid level, then the Newton iteration, and finally the innermost - the Gauss-Seidel iteration. The Gauss-Seidel and Newton iterations do not involve an updating of the boundary conditions. They operate only on the interior of blocks using fixed boundary values.

The quest for time accuracy can thus be a hunt through a multidimensional parameter space formed by the time step, the number of multigrid levels, and the numbers of the various interations. The above results indicate that reducing the time step is not sufficient, at least for problems involving acceleration. Judging from verbal advice of experienced users and sample input files, it is common practice to use five Gauss-Seidel iterations, from one to a few Newton iterations, a few multigrid levels, and from one to a few multigrid cycles. For difficult time dependent problems either the number of multigrid cycles or the number of Newton iterations is increased from one to a few for increased stability or accuracy. Recent experience seems to provide evidence that it is more effective to increase the number of multigrid cycles than to increase the number of Newton iterations.[¶]

Some experiments in varying the other numerical parameters were carried out with $\Delta t = 0.001$. Figure 21 presents time histories of the pressure and friction drags computed with various numbers of multigrid cycles, with $\Delta t = 0.001$, three levels of multigrid, one Newton iteration, and 5 Gauss-Seidel iterations. The pressure drag dominates during the acceleration period but the frictional drag is larger afterward. While the frictional drag is essentially independent of the number of multigrid cycles, the pressure drag is highly dependent on that parameter with noticeable changes upon doubling of the number of iterations up to at least 64. Apparently, it is a much more difficult task to get the pressure on the body converged than to get the velocity profiles at the body surface converged. This is not surprising since the pressure on the body depends on the velocity throughout the flow field and not only adjacent to the body surface. The pressure drag seems to be dominated by the inviscid acceleration reaction with indications of convergence to the expected inverted v shape during the acceleration period and then essentially to zero thereafter. When too few cycles are applied there seems to be a time lag in the pressure drag. The initial rise is not linear in time and the peak occurs much later than the peak in the acceleration at $t = 0.05$. There follows an undershoot around $t = 0.1$ to negative values with following oscillations about zero. It is only when the number of cycles is increased to 32 or 64 that the peak occurs at about the correct time and that the transition from a rapid decrease to essentially zero at $t = 0.1$ is well represented. Note that the number of multigrid cycles necessary may vary in time, with larger numbers needed only during critical periods such as at

[¶]F. Davoudzadeh and L. Taylor, private communications.

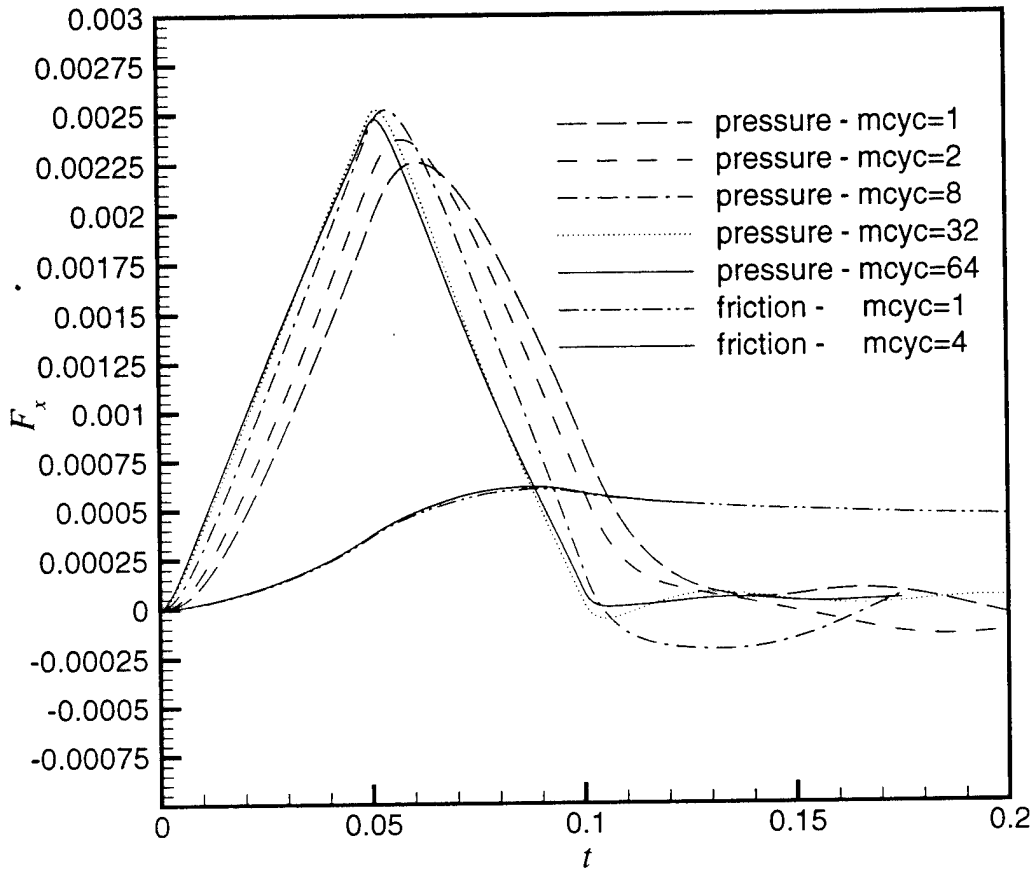


Fig. 21. Computed axial force components for Body 1 accelerated from rest to a constant speed with $\Delta t = 0.001$ and various numbers of multigrid cycles.

the beginning or end of the acceleration.

The drag values can be compared with the added mass coefficient for motion in the axial direction $(X_{\dot{u}})^{\|}$ for Body 1 of about 0.0002.** For the current acceleration this yields an axial force of about 0.002 at $t = 0.05$ rather than the computed value which is about 20% higher. Rerunning the acceleration with the finer grid did not improve this value and this difference warrants further investigation.

As mentioned above, such a rapid acceleration is unrealistic and it is demonstrated below that a more gradual acceleration requires fewer iterations. However, these results still provide valuable experience that might have implications for other types of unsteady movements such as of propellers and appendages.

It is worth noting that cutting the time step or increasing the number of multigrid cycles do not seem to be interchangable. It might be presupposed that the time step could be chosen to be small enough such that one multigrid cycle would suffice for accuracy. The above results, where it is necessary to do multiple multigrid cycles even when the time step is so small that the results are independent of Δt, indicate otherwise. This characteristic could be particular to accelerating body problems.

Figure 22 demonstrates the effect of the number of Newton iterations on the results. Comparison with the effect of the multigrid cycles shows that increasing the Newton iterations has a very similar effect as increasing the multigrid cycles. The results are apparently improved in a very similar manner with either parameter but the pressure lag is corrected slightly better by the additional multigrid cycles (and/or the associated boundary condition updating) than by the Newton iterations. Also the additional Newton iterations seem to yield a somewhat higher (less accurate) peak than the multigrid cycles.

In order to asess separately the contributions of multigrid levels and boundary conditions updating some runs were repeated with multigrid turned off. The solution is then obtained by working only on the original fine grid. In this case a "multigrid" cycle merely adds a boundary condition iteration to what is otherwise a Newton iteration. Figure 23 shows that turning off multigrid results in a greater time lag in the axial pressure force than is obtained with three multigrid levels. This confirms the beneficial effect of multigrid. It is interesting to note though that with multigrid off the pressure force does not undershoot but rather decreases monotonically toward zero after the acceleration period. Without multigrid, an increase in the number of Newton iterations or "multigrid" cycles again leads to a an apparent improvement of the results. As with multigrid, the improvement is seen to be more for two multigrid cycles than for two Newton subiterations. This confirms that the solution benefits from including a boundary condition update in the iteration process. Such a boundary condition effect could be accentuated by the additional (interblock) boundaries of a multiblock grid. An attempt was made to demonstrate such an accentuation by dividing the one-block grid

$^{\|}X_{\dot{u}} = m_a/0.5\rho L^3$ where m_a is dimensional added mass.

**R. Curphey, private communication.

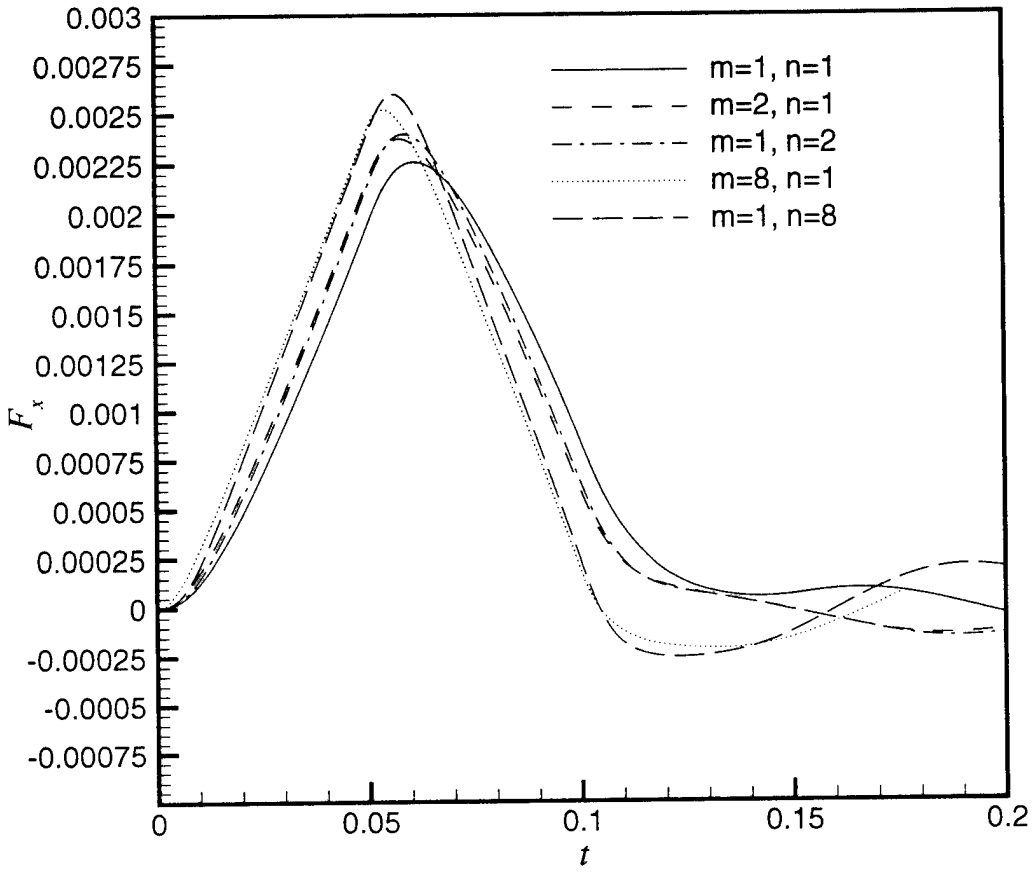


Fig. 22. Computed axial pressure force for Body 1 accelerated from rest to a constant speed with $\Delta t = 0.001$ and various numbers of multigrid cycles (m) and Newton interations (n).

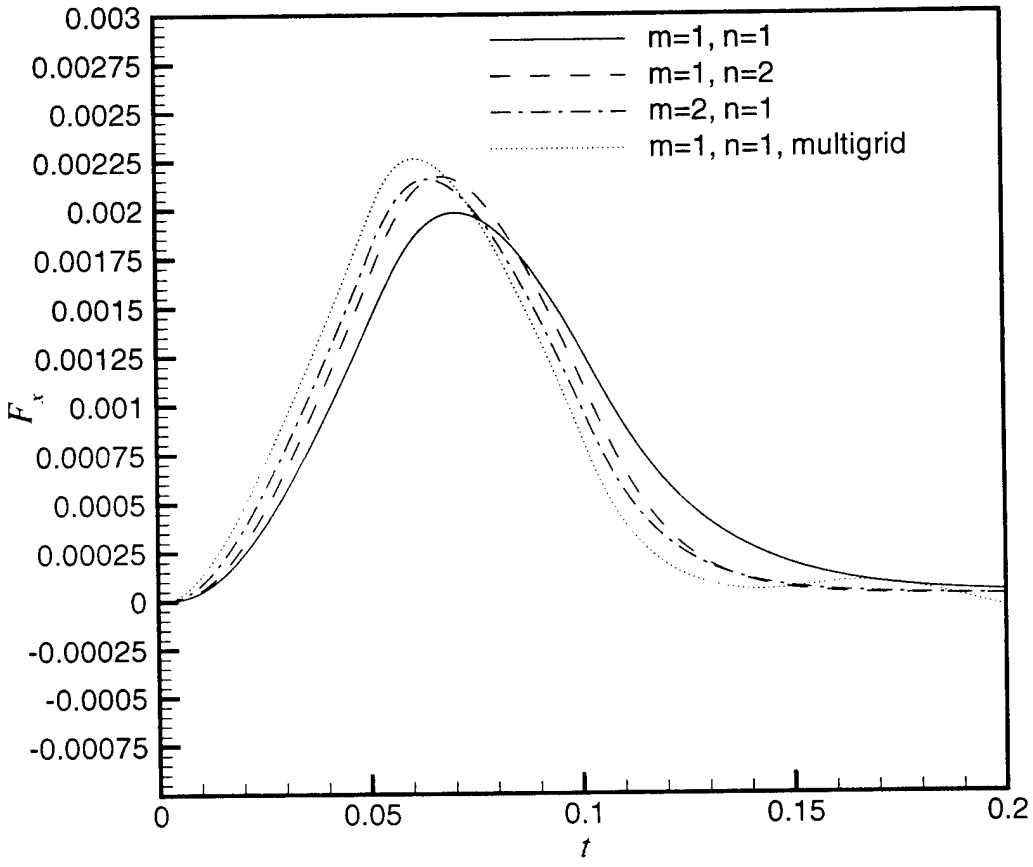


Fig. 23. Computed axial pressure force for Body 1 accelerated from rest to a constant speed without multigrid using various numbers of multigrid cycles (m) and Newton interations (n) and with multigrid for m=1 and n=1.

into two blocks. This was attempted first with a vertical plane block boundary at the middle of the body to form upstream and downstream blocks. No noticeable differences were noted between the one- and two-block solutions with three levels of multigrid, one multigrid cycle and one or two Newton subiterations. The grid was also divided by a horizontal plane block boundary to form upper and lower quadrant blocks. Again the results were essentially the same as for the one-block case except that with two Newton subiterations, the computations abruptly blew up after some time of seemingly normal evolution. With two multigrid cylces and one Newton subiteration, no such instability was encountered.

An experiment was carried out on the dependence of the results on the number of Gauss-Seidel iterations. The acceleration was rerun with one multigrid cycle, one Newton iteration, and one Gauss-Seidel iteration. The results were essentially the same as for five Gauss-Seidel iterations. However, when the number of Newton iterations was increased to two, the computation with only one Gauss-Seidel iteration blew up while with five Gauss-Seidel iterations there was no problem. Thus it seems that some Gauss-Seidel iterations are useful for stability even if they are not needed for accuracy. In any case, Gauss-Seidel iterations seem to be cheap in terms of computational time.

As mentioned above, accuracy can be achieved for less severe accelerations with much less computational work. This is demonstrated in Figure 24 where computed pressure drag histories are presented for an acceleration period of 1.0 instead of 0.1. The time step is accordingly increased to $\Delta t = 0.01$. The pressure drag converges reasonably well with many fewer multigrid cycles than for the more rapid acceleration. The acceleration reaction is well represented with little lag or undershoot and with a small fairly steady viscous contribution to the pressure drag after the acceleration period. (Note that the vertical scale is exaggerated by a factor of 10 compared with the previous figures.) This result is encouraging but the problem still remains of determining the number of iterations necessary to meet desired accuracies for each application. Questions arise such as how accurately the forces need to be computed for computation of an accurate trajectory. Automation of the selection of the number of iterations would be highly desirable.

CONCLUSIONS AND RECOMMENDATIONS

The UNCLE.6DOF submarine maneuvering RANS code was run at the Hydrodynamic and Hydroacoustic Technology Center at CDNSWC. The UNCLE portion of the code was tested on the steady problem of Body 1 at angles of attack. Unsteady test problems included Body 1 accelerating and an appended SUBOFF hull, with turning propeller, moving forward at constant speed. The six degree of freedom portion of the code was only partially tested by running a steady turn as a prescribed trajectory.

UNCLE.6DOF has a host of computational features which give it great promise as a comprehensive submarine maneuvering predictive tool. These features include capabilities for treating full unsteady maneuvers, grid clicking for a rotating propeller, and

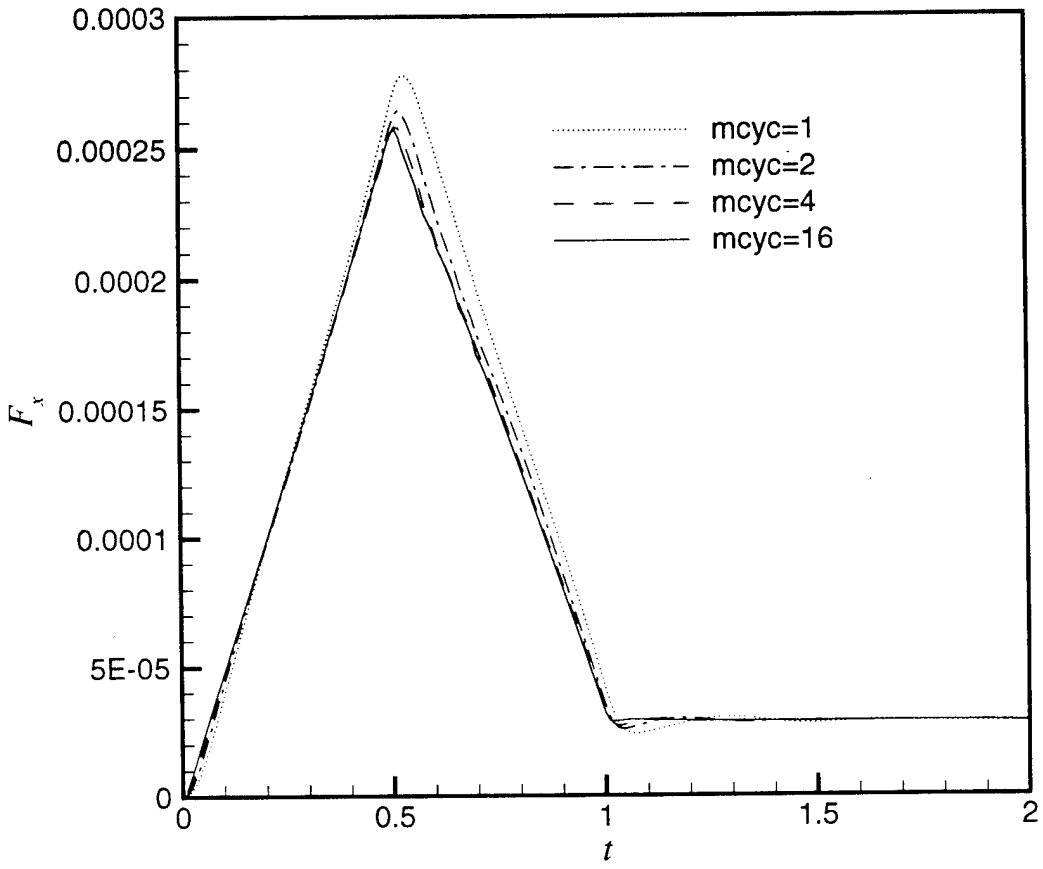


Fig. 24. Computed axial pressure force for Body 1 accelerated slowly from rest to a constant speed with various numbers of multigrid cycles.

multigrid and other techniques for computational efficiency. Its generalized multiblock structure gives it an ability to handle arbitrarily complex configurations. The tests carried out confirm that these features have been successfully integrated into a powerful and general computational system.

Because of its features, CDNSWC is very anxious to put UNCLE.6DOF to use in helping to solve real maneuvering problems. However, even with all of the current computational features, some shortcomings need to be addressed before the code can reach its ultimate capabilities as a powerful tool for submarine maneuvering applications. Necessary improvements relate mainly to more accurate representation of the physics. Other highly desirable upgrades address usability. The need has already been recognized and some improvements are underway at MSU and PSU with support from ONR. For example, one of the more serious limitations of the current code is that the process described in the previous section of using a search parameter within the Baldwin-Lomax model cannot be used for predictive purposes. The determination of the search parameter depends on the existence of measured forces. Since it is unlikely that the Baldwin-Lomax turbulence model can be generalized enough to give it as wide a range of robust accuracy as desired, there is a need for inclusion of more advanced models. To this end PSU has developed a version of the $q - \omega$ model and MSU is currently testing a $k - \epsilon$ model in UNCLE. The CDNSWC submarine maneuvering community feels strongly that such improved models must be included, to substantially improve the accuracy of the force and moment calculations, in any further evaluations.

Work is also underway at MSU to include a capability for moving appendages, a necessity for detailed maneuvering simulations. Additional efforts are aimed at generating parallel versions of UNCLE to accelerate the process of obtaining solutions.

Other improvements should be considered. The results for the accelerating body presented in the previous section indicate that the choice of the numerical parameters depends on the motion under consideration. The addition to the code of an automatic check for convergence to time accuracy and automatic selection of some of the parameters would be a large step toward more dependable usability. The results also indicate the importance of iterating on boundary conditions for convergence and possibly stability. Perhaps boundary condition iterations could be included in the Newton iteration loop with beneficial effect.

The multiblock structure is quite general although there are restrictions on block connectivity that preclude the easy use of a gridding strategy developed in recent years at CDNSWC for efficient placement of grid points for complex appendage configurations. Fortunately other similar codes in use at MSU contain more general block structuring [††] so its inclusion in UNCLE.6DOF should not consume undue resources.

Success has been reported elsewhere on the use of imbedded and Chimera grids. Such techniques should be kept in mind as a possible long-term improvement over

[††]L. Taylor, private communication.

the current grid "clicking" for the propeller and as a possible approach to improving accuracy and generality.

While much of the coding is very general, the author needed to do some minor recoding for the test applications described above. Some of this was due to lines of code which were valid only for the test problem provided with the code. Axis boundary conditions had to be recoded for different grids. In addition, parts of the code were written with the assumption that a propeller is present. While some of the above test cases could be run without a propeller, it is not clear whether more complex unpropelled cases can be run without recoding. Some portions of the code should be recoded for increased generality in order to reduce the need for user recoding. The desirability of eliminating user recoding is likely to increase with the conversion to parallel versions and it is hoped that more advanced versions are developed with as much generality as feasible.

A useful feature of the code is the invocation of noslip, farfield, reflection (symmetry), interblock, and axis boundary conditions through the input. Code could be written to add a slip wall condition, a capability which has proven useful in RANS work at CDNSWC.

There are many versions of UNCLE, at least partly due to the impressive array of improvements that are being pursued simultaneously. Such a situation naturally introduces difficulties in implementing recent advances, in a timely way, into a particular code such as UNCLE.6DOF. In order to facilitate this process, and to aid users who may need to run more than one version of UNCLE, it is recommended that various versions be standardized with respect to input format as much as practical.

Documentation of any sort would be very valuable. Input file descriptions are needed and it is hoped that the appendices of this report are a useful beginning. Other instructions such as gridding guidelines and limitations and how to choose the three reference points for grid movement are necessary if the code is to meet its full potential. In addition, for an otherwise powerful code to be usable it needs to have as many internal checks as possible. Such checks are crucial for saving debugging time as well as helping to prevent erroneous results. An example of such a check already in the code is the search for negative grid volumes. In an attempt by the author to run with a bad grid, the resulting warning messages immediately provided valuable information on the existence and location of a problem. On the other hand, when the author asked for three layers of multigrid with a grid whose dimensions couldn't support that many layers the code seemed to run fine. After it was noticed that the results were incorrect it took some time to find the source of the problem. An immediate warning of the input inconsistency would have been very useful.

Such usability improvements and documentation can be expensive and are not exciting work but it is hoped that parts of the UNCLE team can be identified to perform the work and that support for this excellent software will not end prematurely.

Usability improvements should facilitate further work such as that planned under

the recently approved DOD Challenge Project on Submarine Maneuvering. Numerous cases, including various trajectories, will be computed and recomputed to demonstrate convergence with respect to the various numerical parameters. Results will be compared to available measured data. Successful completion of this process, which depends critically on the improved turbulence modeling discussed above, will yield a computational tool that can facilitate dramatic advances in submarine performance.

ACKNOWLEDGEMENTS

The author hereby acknowledges the contributions of Dr. Farhad Davoudzadeh of PSU, who transferred the code and SUBOFF input to CDNSWC and helped the author get started, and of Dr. Lafayette Taylor of MSU who answered many questions on the use of UNCLE and provided information on the various inputs. Dr. Ming Chang of CDNSWD, Code 5600 also ran UNCLE.6DOF during FY97 and has shared her experience and viewpoints with the author. Mr. Alan Becnel digitized the computed results and measured data previously presented by Sung et al. Computer time was provided by the Navy Hydrodynamic/Hydroacoustic Technology Center.

APPENDIX A: MAKEFILE

The Makefile used for compilation of UNCLE.6DOF on the SGI Power Challenge system at the H/HTC is as follows:

```
OPT3 = -r8 -O3
FPP_FLAGS = -P
OBJ_UNCL = \
main.o\
bci.o bcinfo.o bcj.o bck.o bf.o bfx.o  bfxmap.o\
bfxread.o bfxrestart.o bfxthrust.o cctf.o\
cvmgp.o cvmgz.o deqlu.o dogsm.o dogsp.o\
eddyj.o eddyk.o eigenv.o flux.o force_prp.o fwakej.o\
fwakek.o gridv.o ic.o\
icvmgm.o icvmgm2.o icvmgp.o icvmgt.o icvmgz.o\
intrface.o loadqi.o ludec.o lusol.o mapping.o\
metric.o nd.o newgrid.o newgridu.o qftc.o qrql.o\
rcvmgm.o rcvmgp.o rcvmgt.o rcvmgz.o \
resid.o restr.o rftc.o rlvecs.o rsin.o rsout.o\
setdr.o smooth.o step.o vnd.o xgr.o xyz.o\
force.o propforce.o  prescribe.o

OBJ_6DOF = \
atanx.o in_bf.o sixdof.o initveh.o bf_in.o rhs.o\
convertfm.o rkm.o cputime_ibm.o move.o mtinv.o dyngrd.o
OBJ_PVAR = pvar.o pva.o

Ex.u6   : precision.inc parameter.inc one.inc $(OBJ_UNCL) $(OBJ_6DOF) $(OBJ_PVAR)
f77 $(OPT3) -o Ex.u6   $(OBJ_UNCL) $(OBJ_6DOF) $(OBJ_PVAR)
size Ex.u6
$(OBJ_UNCL) : precision.inc parameter.inc one.inc
/lib/cpp $(FPP_FLAGS) -I. < $*.f >DBG/$*.f
f77 -c $(OPT3) DBG/$*.f
$(OBJ_6DOF) : precision.inc parameter.inc one.inc
/lib/cpp $(FPP_FLAGS) -I. < $*.f >DBG/$*.f
f77 -c $(OPT3) DBG/$*.f
$(OBJ_PVAR) : precision.inc parameter.inc one.inc
/lib/cpp $(FPP_FLAGS) -I. < $*.f >DBG/$*.f
f77 -c $(OPT3) DBG/$*.f

clean:
\rm $(OBJ_UNCL) $(OBJ_6DOF) $(OBJ_PVAR)
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B: INPUT FILE UNCLE.INP

The file UNCLE.INP contains most of the numerical parameters. The following example was received from PSU with the propelled SUBOFF geometry. Descriptive comments, denoted by percent signs, have been added.

```
 UNCLE_MB_6DOF ALGORITHM PARAMETERS
 NT          NS1         NS2         IRS         INITRU     IQUART
 200         1           200         1           0          1
%
% NT       - number of cycles (time steps or pseudo-time steps) to be run
% NS1      - number of cycles between residual information output
% NS2      - number of cycles between writing of full restart file
% IRS      - =0 to start a problem from scratch
%            =1 to do a restart run (restart file needed)
% INITRU   - =1 to restart from an uncle generated restart file
%            =0 to restart from an uncle_6dof generated restart file
% IQUART   - propeller rotation/boat direction quadrant
%
 MCYC        MOTION      MDMP
 1           0           0
%
% MCYC     - number of multigrid cycles per time step
% MOTION   - =1 for using the 6dof motion code
%            =0 for uncle part alone with straight ahead motion at constant speed
%            =2 for uncle with prescribed trajectory (in author's version)
% MDMP     - =1 to rotate coordinate axes to match Navy maneuvering convention
%            =0 to keep coordinates from restart file
%
 IDYN        ISTATE      KEYCONT
 1           0           1
%
% IDYN     - =1 all or part of the grid is moving
% ISTATE   - =0 for an unsteady problem
%            =1 for a steady state problem
% KEYCONT  - used for controlling the direct writing of plot files
%
 CFL(-DT)    ISGS        IFREQ       NSUB
 -0.0006782  5           9999        1
%
% CFL      - <0 for automatic computing of time step from DTDT and NTREV
%            must be so if MOTION = 1
```

```
%           >0 to use specified value as the time step
% ISGS     - number of gauss-seidel interations per Newton iteration
% IFREQ    - frequency of updating solution matrix operators
%           (hardwired to 1  for this code)
% NSUB     - number of Newton subiterations at each multigrid level
%
 DTDTB      DTDTE      NTDRPM     NTREV      NBROT1     NBROT2
 72.00      72.00      240        160        37         51
%
% DTDTB    - rotational frequency of propeller at the beginning of the run
%           (radians per unit of dimensionless time)
% DTDTE    - rotational frequency of propeller NTDRPM cycles later
% NTDRPM   - number of time steps over which the change in prop rate is spread
% NTREV    - number of time steps per propeller rotation period
% NBROT1   - the first block of a group of rotating blocks
% NBROT2   - the last block of a group of rotating blocks
%
 LIMIT      ORDER      BETA       LTSDG
 0          3.         5.         0
%
% LIMIT    - refers to limiters - not used
% ORDER    - spatial accuracy - nearly always 3rd order
% BETA     - pseudo-compressibility factor - usually in the range of 5-10
% LTSDG    - =1 to use spatially variable time steps to accelerate
%                converenge of steady solutions
%           =0 to use spatially constant time steps for time accuracy
%
 EPI        EPJ        EPK
 0.5        0.5        0.5
%
% EPI      - multigrid correction smoothing parameter in the i-direction
% EPJ      - multigrid correction smoothing parameter in the j-direction
% EPK      - multigrid correction smoothing parameter in the k-direction
%
 NRKM       IWRT
 1          100000
%
% NRKM     - number of sub time steps used to move the grid in 6DOF code
% IWRT     - number of cycles between the writing of a visualization file
%           in PLOT3D format
%
```

```
 NBRS        NBRE
 25          51
%
% NBRS     - block number of the first block in the first blade row
% NBRE     - block number of the last block in the last blade row
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C: INPUT FILE BC.INFO.UMG

The file BC.INFO.UMG contains the UNCLE boundary condition information. The following example was received from PSU with the propelled SUBOFF geometry. The author has added comments in the block 1 section, to the best of his understanding at the time of the writing of this report.

```
1,0,1,0,1,1,0.0           / n,nbr,nblj,nblk,nbld,mblk,dtdt
%
% n    - the number of the block
% nbr  - the number of the axial blade row in which the block resides
%        = 1 for stern appendage (stator) blocks
%        = 2 for propeller blocks
% nblj - the number of the radial ring in which the block resides
%        = 1 for the inner ring (near the body)
%        = 2 for the middle ring
%        = 3 for the outer ring
% nblk - the number of the azimuthal wedge in which the block resides
%        = 1 to 4 for the stern appendage blocks
%        = 1 to 5 for the propeller blocks
% nbld - number of blades or vanes
% mblk - number of blocks in the blade row
% dtdt - not used
%
0,0                    / ids,ide
%
% ids  - number of grid surfaces in clicking region near i = 1
% ide  - number of grid surfaces in clicking region near i = ie
%
1,1,2,                /jedy,jbnd1,jbnd2
1,1,1,                /kedy,kbnd1,kbnd2
%
% jedy - = 1 if eddy viscosity is to be computed off j=constant wall
%(jbnd1,jbnd2)- = (1,1) if wall at j = 1 only
%        = (1,2) if walls at both j = 1 and je
%        = (2,2) if wall at j = je only
% kedy - = 1 if eddy viscosity is to be computed off k=constant wall
%(kbnd1,kbnd2)- similar to (jbnd1,jbnd2)
%
1,-13,9,-5,-4,-2,     / ibc1,ibc2,jbc1,jbc2,kbc1,kbc2
%
% ibc1 - boundary condition to apply at i = 1
```

```
% ibc2 - boundary condition to apply at i = ie
% jbc1 - boundary condition to apply at j = 1
% jbc2 - boundary condition to apply at j = je
% kbc1 - boundary condition to apply at k = 1
% kbc2 - boundary condition to apply at k = ke
% Some precoded boundary conditions
%    0  - farfield boundary condition
%    1  - axis boundary condition - may need to be recoded
%    9  - noslip
%    12 - reflection of v velocity
%    14 - reflection of w velocity
%    -n - interblock connection to block n
%
%    Note that interblock boundaries are currently considered to be
%    a matching of either i=ie to i=1, j=je to j=1 or k=ke to k=1.
%    Anything else will require writing of code.
%
1,49,1,41,25,41,1,25,   /nisj1,niej1,nksj1,nkej1,nisj2,niej2,nksj2,nkej2
%
% These are numbers which can be used to define sections of j and
% k = constant boundaries which are noslip.  They will override the
% above boundary conditions on those sections.
% For example block 1 of this grid extends
% from the nose to the trailing edge of the sail.  Thus the k=1
% boundary is partly interblock and partly noslip.  The k=1 boundary
% is first specified as connecting to block 4 by setting kbc1=-4
% as if it were all interblock.  Then the section running from i=17 to
% i=49 and j=1 to j=17 is specified as being noslip through setting
% nisk1,niek1,njsk1, and njek1 below.  If the complete boundary is noslip
% then these variables must reflect that and the original boundary
% condition setting is ignored.  If the original boundary condition
% specification - other than noslip - is to be used, then it is sufficient
% that the four variables corresponding to that boundary be set equal to 1.
% nisj1 - beginning i coordinate of segment of j=1 boundary
% niej1 - ending i coordinate of segment of j=1 boundary
% nksj1 - beginning k coordinate of segment of j=1 boundary
% nkej1 - ending k coordinate of segment of j=1 boundary
% nisj2 - beginning i coordinate of segment of j=je boundary
% niej2 - ending i coordinate of segment of j=je boundary
% nksj2 - beginning k coordinate of segment of j = je boundary
% nkej2 - ending k coordinate of segment of j = je boundary
```

```
%
17,49,1,17,1,1,1,1,     /nisk1,niek1,njsk1,njek1,nisk1,niek2,njsk2,njek2
%
% nisk1 - beginning i coordinate of segment of k=1 boundary
% niek1 - ending i coordinate of segment of k=1 boundary
% njsk1 - beginning j coordinate of segment of k=1 boundary
% njek1 - ending j coordinate of segment of k=1 boundary
% nisk2 - beginning i coordinate of segment of k=ke boundary
% niek2 - ending i coordinate of segment of k=ke boundary
% njsk2 - beginning j coordinate of segment of k=ke boundary
% njek2 - ending j coordinate of segment of k=ke boundary
%
%end of info for block 1
%
%
2,0,1,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
1,1,1,
0,1,1,
1,-14,9,-6,-1,-3,
1,49,1,41,1,1,1,1,
1,1,1,1,1,1,1,1,
3,0,1,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
1,1,1,
0,1,1,
1,-15,9,-7,-2,-4,
1,49,1,41,1,1,1,1,
1,1,1,1,1,1,1,1,
4,0,1,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
1,1,2,
1,2,2,
1,-16,9,-8,-3,-1,
1,49,1,41,25,41,17,41,
1,1,1,1,17,49,1,17,
5,0,2,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
1,1,1,
1,1,1,
1,-17,-1,-9,-8,-6,
```

```
25,41,1,25,1,1,1,1,
17,49,1,17,1,1,1,1,
6,0,2,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
0,1,1,
0,1,1,
1,-18,-2,-10,-5,-7,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
7,0,2,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
0,1,1,
0,1,1,
1,-19,-3,-11,-6,-8,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
8,0,2,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
1,1,1,
1,2,2,
1,-20,-4,-12,-7,-5,
25,41,17,41,1,1,1,1,
1,1,1,1,17,49,1,17,
9,0,3,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
0,1,1,
0,1,1,
1,-21,-5,0,-12,-10,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
10,0,3,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
0,1,1,
0,1,1,
1,-22,-6,0,-9,-11,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
11,0,3,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
0,1,1,
0,1,1,
```

```
1,-23,-7,0,-10,-12,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
12,0,3,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                     / ids,ide
0,1,1,
0,1,1,
1,-24,-8,0,-11,-9,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
13,0,1,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                     / ids,ide
1,1,1,
0,1,1,
-1,-25,9,-17,-16,-14,
1,49,1,41,1,1,1,1,
1,1,1,1,1,1,1,1,
14,0,1,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                     / ids,ide
1,1,1,
0,1,1,
-2,-26,9,-18,-13,-15,
1,49,1,41,1,1,1,1,
1,1,1,1,1,1,1,1,
15,0,1,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                     / ids,ide
1,1,1,
0,1,1,
-3,-27,9,-19,-14,-16,
1,49,1,41,1,1,1,1,
1,1,1,1,1,1,1,1,
16,0,1,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                     / ids,ide
1,1,1,
0,1,1,
-4,-28,9,-20,-15,-13,
1,49,1,41,1,1,1,1,
1,1,1,1,1,1,1,1,
17,0,2,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                     / ids,ide
0,1,1,
```

```
0,1,1,
-5,-29,-13,-21,-20,-18,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
18,0,2,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
0,1,1,
0,1,1,
-6,-30,-14,-22,-17,-19,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
19,0,2,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
0,1,1,
0,1,1,
-7,-31,-15,-23,-18,-20,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
20,0,2,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
0,1,1,
0,1,1,
-8,-32,-16,-24,-19,-17,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
21,0,3,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
0,1,1,
0,1,1,
-9,-33,-17,0,-24,-22,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
22,0,3,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
0,1,1,
0,1,1,
-10,-34,-18,0,-21,-23,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
23,0,3,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                    / ids,ide
```

```
0,1,1,
0,1,1,
-11,-35,-19,0,-22,-24,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
24,0,3,0,1,1,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,0                     / ids,ide
0,1,1,
0,1,1,
-12,-36,-20,0,-23,-21,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
25,1,1,1,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                     / ids,ide
1,1,1,
1,1,2,
-13,25,9,-29,-28,-26,
1,33,1,41,1,1,1,1,
1,17,1,17,1,17,1,17,
26,1,1,2,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                     / ids,ide
1,1,1,
1,1,2,
-14,29,9,-30,-25,-27,
1,33,1,41,1,1,1,1,
1,17,1,17,1,17,1,17,
27,1,1,3,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                     / ids,ide
1,1,1,
1,1,2,
-15,29,9,-31,-26,-28,
1,33,1,41,1,1,1,1,
1,17,1,17,1,17,1,17,
28,1,1,4,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                     / ids,ide
1,1,1,
1,1,2,
-16,29,9,-32,-27,-25,
1,33,1,41,1,1,1,1,
1,17,1,17,1,17,1,17,
29,1,2,1,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
```

```
0,4                    / ids,ide
0,1,1,
0,1,1,
-17,29,-25,-33,-32,-30,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
30,1,2,2,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                    / ids,ide
0,1,1,
0,1,1,
-18,29,-26,-34,-29,-31,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
31,1,2,3,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                    / ids,ide
0,1,1,
0,1,1,
-19,29,-27,-35,-30,-32,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
32,1,2,4,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                    / ids,ide
0,1,1,
0,1,1,
-20,29,-28,-36,-31,-29,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
33,1,3,1,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                    / ids,ide
0,1,1,
0,1,1,
-21,29,-29,0,-36,-34,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
34,1,3,2,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                    / ids,ide
0,1,1,
0,1,1,
-22,29,-30,0,-33,-35,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
```

```
35,1,3,3,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                     / ids,ide
0,1,1,
0,1,1,
-23,29,-31,0,-34,-36,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
36,1,3,4,4,4,0.0          / n,nbr,nblj,nblk,nbld,mblk,dtdt
0,4                     / ids,ide
0,1,1,
0,1,1,
-24,29,-32,0,-35,-33,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
37,2,1,1,5,5,77.2         / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                     / ids,ide
1,1,1,
1,1,2,
25,0,1,-42,-41,-38,
1,49,1,33,1,1,1,1,
17,33,1,17,17,33,1,17,
38,2,1,2,5,5,77.2         / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                     / ids,ide
1,1,1,
1,1,2,
29,0,1,-43,-37,-39,
1,49,1,33,1,1,1,1,
17,33,1,17,17,33,1,17,
39,2,1,3,5,5,77.2         / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                     / ids,ide
1,1,1,
1,1,2,
29,0,1,-44,-38,-40,
1,49,1,33,1,1,1,1,
17,33,1,17,17,33,1,17,
40,2,1,4,5,5,77.2         / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                     / ids,ide
1,1,1,
1,1,2,
29,0,1,-45,-39,-41,
1,49,1,33,1,1,1,1,
```

```
17,33,1,17,17,33,1,17,
41,2,1,5,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
1,1,1,
1,1,2,
29,0,1,-46,-40,-37,
1,49,1,33,1,1,1,1,
17,33,1,17,17,33,1,17,
42,2,2,1,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
0,1,1,
0,1,1,
29,0,-37,-47,-46,-43,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
43,2,2,2,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
0,1,1,
0,1,1,
29,0,-38,-48,-42,-44,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
44,2,2,3,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
0,1,1,
0,1,1,
29,0,-39,-49,-43,-45,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
45,2,2,4,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
0,1,1,
0,1,1,
29,0,-40,-50,-44,-46,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
46,2,2,5,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
0,1,1,
0,1,1,
29,0,-41,-51,-45,-42,
```

```
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
47,2,3,1,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
0,1,1,
0,1,1,
29,0,-42,0,-51,-48,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
48,2,3,2,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
0,1,1,
0,1,1,
29,0,-43,0,-47,-49,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
49,2,3,3,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
0,1,1,
0,1,1,
29,0,-44,0,-48,-50,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
50,2,3,4,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
0,1,1,
0,1,1,
29,0,-45,0,-49,-51,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
51,2,3,5,5,5,77.2          / n,nbr,nblj,nblk,nbld,mblk,dtdt
4,0                      / ids,ide
0,1,1,
0,1,1,
29,0,-46,0,-50,-47,
1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D: INPUT FILE VEHICLE.INP

The file VEHICLE.INP contains the physical parameters of the submarine and the definition of its initial location and motion as well as the physical characteristics of the water and its farfield motion. The following file was received from PSU with the propelled SUBOFF geometry. Comments have been added by the author.

```
-->REFERENCE QUANTITIES:  *** SUBOFF CAPTIVE MODEL ***
rhoref      uref        dref        xmuref      gc
1.937       10.972      14.2917     2.36d-05    32.174
%
% The first four of these quantities determine the Reynolds number.
% For UNCLE-only runs (no computation of motion), these parameters are
% used only in computing the Reynolds number and they can be set in any
% way that results in the desired value of that parameter.
%
% rhoref - water density
% uref   - reference velocity
% dref   - reference length
% xmuref - dynamic viscosity
% gc     - gravitational acceleration
%
-->FAR-FIELD CHARACTERISTICS:  *** MOTION = 0 ***
vinf        alpha       chi         phi
0.0         0.0         0.0         0.0
%
% These quantities can be used to impose a farfield flow.
% They can be used with idyn = 0 (in UNCLE.INP) to do steady
% flow about a body at angles of attack.
%
% vinf   - dimensionless farfield velocity
% alpha  - pitch angle relative to farfield flow in degrees
% chi    - yaw angle relative to farfield flow
% phi    - roll angle relative to farfield flow
%
-->RIGID BODY CHARACTERISTICS: *** MOTION = 1 ***
weight      xg          yg          zg
1557.0      -1.327d-02  0.000d-00   0.000d-00
%
% weight  - weight
% xg      - x-coordinate of center of gravity (dimensional)
% yg      - y-coordinate of center of gravity
```

```
% zg      - z-coordinate of center of gravity
%
buoy        xb          yb          zb
1557.0      -1.327d-02  0.000d+00   -6.70d-03
%
% buoy    - buoyancy
% xb      - x-coordinate of center of buoyancy
% yb      - y-coordinate of center of buoyancy
% zb      - z-coordinate of center of buoyancy
%
xix         xiy         xiz
15.84       560.0       560.0
%
% xix     - mass moment of inertia about x-axis
% xiy     - mass moment of inertia about y-axis
% xiz     - mass moment of inertia about z-axis
%
xixy        xixz        xiyz
0.000d+00   0.000d+00   0.000d+00
%
% xixy    - xy cross product moment of inertia
% xixz    - xz cross product moment of inertia
% xiyz    - yz cross product moment of inertia
%
-->BODY AXIS STATE VARIABLES: *** IRS = 0 ***
uvel        vvel        wvel
0.0         0.0         0.0
%
% uvel    - initial x-velocity of the body (dimensional)
% vvel    - initial y-velocity of the body
% wvel    - initial z-velocity of the body
%
prot        qrot        rrot
0.0         0.0         0.0
%
% prot    - initial rolling angular velocity
% qrot    - initial pitching angular velocity
% rrot    - initial yawing angular velocity
%
-->INERTIAL FRAME POSITION & ORIENTATION *** IRS = 0 ***
xpos        ypos        zpos
```

```
6.604        0.0          0.0
%
% xpos      - initial body frame x-coordinate of inertial frame origin
% ypos      - initial body frame y-coordinate of inertial frame origin
% zpos      - initial body frame z-coordinate of inertial frame origin
%
phi          theta        psi        <---[radians]
0.000d+00    0.000d+00    0.000d+00
%
% phi       - initial roll angle
% theta     - initial pitch angle
% psi       - initial yaw angle
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E: PARAMETER.INC INCLUDE FILE

Major dimensions are specified through the PARAMETER.INC FORTRAN include file. The include statements in the subroutines are written in C and a C preprocessor is used at the time of compilation. The following example was provided with the code to run the SUBOFF problem. Additional comments have been added at the bottom.

```
C
C     THIS FILE CONTAINS THE UNCLE'S ARRAY DIMENSIONS
C               To minimize memory usage
C     *** EACH PROBLEM MUST BE DIMENSIONED EXACTLY!! ***
C       NNI = NO. OF GRID PTS. IN I-DIRECTION
C       NNJ = NO. OF GRID PTS. IN J-DIRECTION
C       NNK = NO. OF GRID PTS. IN K-DIRECTION
C       NB  = NO. OF GRID BLOCKS
C       MG  = NO. OF MULTIGRID LEVELS
C       IS  = 0 -> NO COARSENING IN I; 1 -> COARSEN IN I
C       JS  = 0 -> NO COARSENING IN J; 1 -> COARSEN IN J
C       KS  = 0 -> NO COARSENING IN K; 1 -> COARSEN IN K
C
      parameter(nni=65,nnj=17,nnk=41,nb=51)
      parameter(mg=3,is=1,js=1,ks=1)
      parameter(ni1=nni+1,nj1=nnj+1,nk1=nnk+1)
      parameter(ni2=nni+2,nj2=nnj+2,nk2=nnk+2)
      parameter(mbr=2,njbr=49,nkbr=160,mif=1)

C nni  - greater than or equal to the largest i-dimension of all the
C        blocks
C nnj  - greater than or equal to the largest j-dimension of all the
C        blocks
C nnk  - greater than or equal to the largest k-dimension of all the
C        blocks
C mbr  - greater than or equal to the number of blade rows
C njbr - greater than or equal to the number of grid points in the
C        j-direction for the blade rows
C nkbr - greater then or equal to the number of grid points in the
C        k-direction for the blade rows
C mif  - greater than or equal to the number of blade row interfaces
         (the number of blade rows minus 1)
```

Unless multigrid is turned off in the corresponding direction, the actual dimensions of each block, which are specified in the grid file, must be one greater than a number which yields an integer when divided by 2^{mg-1}.

Upon execution of the code enough memory is reserved to store all of the primary variables at (nni x nnj x nnk x nb) locations. Note that this will usually be more locations than necessary since many blocks will usually have one or more dimensions which are less than the maximum dimensions. In addition, scratch space is set aside which is proportional in size to (nni x nnj x nnk). For a problem with a small number of blocks, this scratch space can be significant. Thus, for such a problem it is possible to cut the memory requirement significantly by subdividing blocks and reducing the maximum block dimensions.

REFERENCES

1. Zierke, W.C. (ed.), "A Physics-Based Means of Computing the Flow Around a Maneuvering Underwater Vehicle," The Pennsylvania State University, Applied Research Laboratory, Technical Report No. TR 97-002, January 1997.

2. Taylor, L.K and Whitfield, D.L., "Unsteady Three-Dimensional Euler and Navier-Stokes Solver for Stationary and Dynamic Grids," AIAA-91-1650, June 1991.

3. Smith, N.S., and Watkinson, K.W., "A Six-Degree-of-Freedom Simulation Program for Underwater Vehicles with Significant Vortical Flow Effects," V.C.T. Report No. 3 (User's Manual for TRJv), Vehicle Control Technologies, Inc., August 1994.

4. Busby, J.A., "Unsteady 3-D Incompressible Flow Interaction in Multiple-Blade-Row Turbomachinery," Ph.D. Dissertation, Mississippi State University, Mississippi State, MS, May 1997.

5. Busby, J.A., "Evaluation of the Mississippi State University Computational Fluid Dynamics Code (UNCLE), CDNSWC Hydromechanics Directorate R&D Report CRDKNSWD/HD-1318-01, Sept. 1996.

6. Huang, T.T., Santelli, and Belt, G., "Stern Boundary-Layer Flow on Axisymmetric Bodies", Proceedings of the 12th Symposium on Naval Hydrodynamics, National Academy of Sciences, Washington, DC, pp. 127-157, 1978.

7. Sung, C.H., Fu, T.C., Griffin, M.J., and Huang, T.T., "Validation of Incompressible Flow Computation of Forces and Moments on Axisymmetric Bodies Undergoing Constant Radius Turning," Twenty-First Symposium on Naval Hydrodynamics. Trondheim, Norway, June 23-28, 1996.

8. Degani, D. and Schiff, L.B., "Computation of Turbulent Supersonic Flows Around Pointed Bodies Having Crossflow Separation," *Journal of Computational Physics*, Vol. 66. No. 1, pp. 173-196, 1986.

9. Taylor, L.K., Busby, J.A., Jiang, M.Y., Arabashi, A., Sreenivas, K., and Whitfield, D.L., "Time Accurate Incompressible Navier-Stokes Simulation of the Flapping Foil Experiment," Proceedings of the Sixth International Converence on Numerical Ship Hydrodynamics, University of Iowa, August 1993.

10. Sheng, C., Taylor, L.K., and Whitfield, D.L., "A Multigrid Algorithm for Unsteady Incompressible Euler and Navier-Stokes Flow Computations," Sixth International Symposium in Computational Fluid Dynamics, Lake Tahoe, Nevada, September, 1995.

11. Lugt, H.J. and Haussling, H.J., "The Acceleration of Thin Cylindrical Bodies in a Viscous Fluid," *Journal of Applied Mechanics*, Vol. 100, No. 1, pp. 1-6, 1978.