DAVID W. TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CENTER

Bethesda, Md. 20084

A NONLINEAR MATHEMATICAL MODEL OF MOTIONS OF A PLANING BOAT IN REGULAR WAVES

by

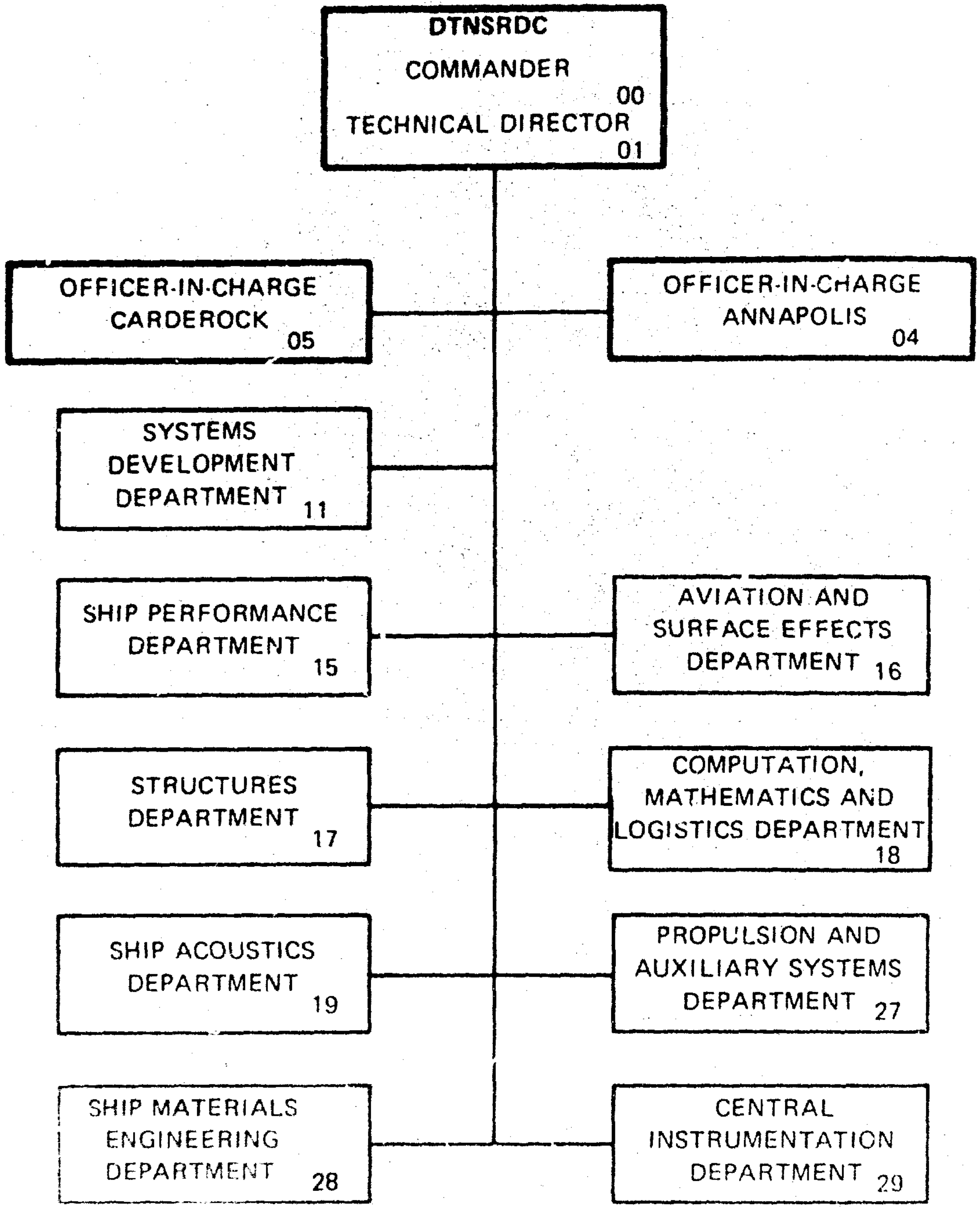
Ernest E. Zarnick

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

SHIP PERFORMANCE DEPARTMENT
RESEARCH AND DEVELOPMENT REPORT

DTNSRDC-78/[illegible]

MAJOR DTNSRDC ORGANIZATIONAL COMPONENTS



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
(14) DTNSRDC-78/032

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
(6) A NONLINEAR MATHEMATICAL MODEL OF MOTIONS OF A PLANING BOAT IN REGULAR WAVES.

5. TYPE OF REPORT & PERIOD COVERED

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
(10) Ernest E./Zarnick

8. CONTRACT OR GRANT NUMBER(s)
(16) F43421, SR02301

9. PERFORMING ORGANIZATION NAME AND ADDRESS
David W. Taylor Naval Ship Research
and Development Center
Bethesda, Maryland 20084

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
(17) ZF 43 421001, SR0230101
~~Work Unit 1-1500-100~~

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Sea Systems Command (SEA 035)
Washington, D.C. 20362

12. REPORT DATE
(11) Mar[redacted]78

13. NUMBER OF PAGES
86 (12) 81 P.

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Planing Boat Motions
Hydrodynamic Impact
Small Boat Worthiness
Nonlinear Ship Motions in Waves

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A nonlinear mathematical model has been formulated of a craft having a constant deadrise angle, planing in regular waves, using a modified low-aspect-ratio or strip theory. It was assumed that the wavelengths would be large in comparison to the craft length and that the wave slopes would be small. The coefficients in the equations of motion were determined by a combination of theoretical and empirical relationships. A simplified version for the case of a craft or model being towed at constant speed was programed for computations on a digital computer, and the results were compared with existing experimental data.

(Continued on reverse side)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

[illegible] JOB

(Block 20 continued)

Comparison of computed pitch and heave motions and phase angles with corresponding experimental data was remarkably good. Comparison of bow and center of gravity vertical accelerations was fair to good.

TABLE OF CONTENTS

LIST OF FIGURES

Page

NOTATION

A	Mass matrix
A_R	Section area
a	Correction factor for buoyancy force
b	Half-beam of craft
$C_{D,c}$	Crossflow drag coefficient
C_Δ	Load coefficient $\Delta/\rho g(2b)^3$
C_λ	Wavelength coefficient $L/\lambda\,[C_\Delta/(L/2b)^2]^{1/3}$
D	Friction drag force
F_x	Total hydrodynamic force in x direction
F_z	Total hydrodynamic force in z direction
F_θ	Total hydrodynamic moment about pitch axis
f	Two-dimensional hydrodynamic force
g	Acceleration of gravity
H	Wave height, crest to trough
h	Vertical submergence of point below free surface
h_z	Double amplitude of heave
I	Pitch moment of inertia
I_a	Added pitch, moment of inertia
k	Wave number
k_a	Two-dimensional added-mass coefficient
L	Hull length
LCG	Longitudinal center of gravity, percent of L
M	Mass of craft
M_a	Added mass of craft

m_a	Sectional (two-dimensional) added mass
N	Hydrodynamic force normal to baseline
r	Wave elevation $r = r_o \cos(kx+\omega t)$
r_o	Wave amplitude
U	Relative fluid velocity parallel to baseline
V	Relative fluid velocity normal to baseline
$V/\sqrt{L}$	Speed-to-length ratio in knots/ft$^{1/2}$
W	Weight of craft
w_z	Vertical component of wave orbital velocity
$\dot{w}_z$	Vertical component of wave orbital acceleration
x	Fixed horizontal coordinate
$\bar{x}$	Vector of state variables
$\dot{x}_{CG}$	Surge velocity
$\ddot{x}_{CG}$	Surge acceleration
x_{CG}	Surge displacement
z	Fixed vertical coordinate
$\dot{z}_{CG}$	Heave velocity
$\ddot{z}_{CG}$	Heave acceleration
z_{CG}	Heave displacement
β	Deadrise angle
Δ	Hull displacement W
ζ	Body coordinate normal to baseline
λ	Wavelength
θ	Pitch angle
$\dot{\theta}$	Pitch angular velocity

$\ddot{\theta}$	Pitch angular acceleration
θ_p	Double amplitude of pitch
ξ	Body coordinate parallel to baseline
ρ	Density of water
ω	Wave frequency
ℓ	Wetted length

ABSTRACT

A nonlinear mathematical model has been formulated of a craft having a constant deadrise angle, planing in regular waves, using a modified low-aspect-ratio or strip theory. It was assumed that the wavelengths would be large in comparison to the craft length and that the wave slopes would be small. The coefficients in the equations of motion were determined by a combination of theoretical and empirical relationships. A simplified version for the case of a craft or model being towed at constant speed was programed for computations on a digital computer, and the results were compared with existing experimental data. Comparison of computed pitch and heave motions and phase angles with corresponding experimental data was remarkably good. Comparison of bow and center of gravity vertical accelerations was fair to good.

ADMINISTRATIVE INFORMATION

This investigation was authorized by the Naval Sea Systems Command with initial funding under Task Area SR-023-0101 and completion under Task Area ZF-43-421001.

INTRODUCTION

Computer programs for estimating the motions of displacement ships in waves for all headings and speeds have been in existence for some time. Comparable computational schemes for planing craft do not exist except in limited and restricted cases. A program for planing craft would be quite useful to the small craft designer, providing a means for systematically exploring the effects of numerous design variations on performance of the craft in waves. With minor modification, the program could also be used to examine the merits of a hybrid craft design, e.g., a combination of planing craft and hydrofoil.

Predicting the motions of a planing craft in wave's is by no means a simple problem. The analytical description of a high-speed craft, planing in waves, involves several different types of flow phenomena, including planing; hydrodynamic impact, and, to a lesser extent, surface wave generation and hydrostatics. Also, the mathematics tend to become nonlinear rapidly as the motion increases or, like the real craft, can in some instances exhibit large instabilities such as porpoising.

Development of a computer program that would take into account all of the previously described factors and would be applicable for a wide range of speed and wave conditions requires a careful and systematic study in several stages with appropriate verification at each stage. To lay the foundation for such a general program, a simpler problem has been formulated in this report with potential for expansion and generalization to the more complicated case. The simpler problem is that of a V-shaped prismatic body with hard chines and constant deadrise planing at high speed in regular head waves.

The mathematical formulation is analogous to low-aspect-ratio wing theory with provisions for including hydrodynamic impact loads, essentially a strip theory. Surface wave generation and forces associated with unsteady circulatory flow are neglected, and the flow is treated as quasi-steady. The mathematical formulation is an empirical synthesis of several theoretically derived flows describing the overall craft hydrodynamics. Wave input is restricted to monochromatic linear deepwater waves with moderate wavelengths and low wave slopes.

MATHEMATICAL FORMULATION

GENERAL

Consider a fixed coordinate system (x,z) (Figure 1) with x axis in the undisturbed free surface, pointing in the direction of craft travel, and the z axis, pointing downward. If the motions of the craft are restricted to pitch θ, heave z_{CG}, and surge x_{CG}, the equation of motions can be written as

$$\begin{aligned} M\ddot{x}_{CG} &= T_x - N \sin \theta - D \cos \theta \\ M\ddot{z}_{CG} &= T_z - N \cos \theta + D \sin \theta + W \\ I\ddot{\theta} &= Nx_c - Dx_d + Tx_p \end{aligned} \tag{1}$$

where M is mass of craft
- I is pitch moment of inertia of craft
- N is hydrodynamic normal force
- D is friction drag
- W is weight of craft
- T_x is thrust component in x direction
- T_z is thrust component in z direction
- x_c is distance from center of gravity (CG) to center of pressure for normal force
- x_d is distance from CG to center of action for friction drag force
- x_p is moment arm of thrust about CG.

Equation (1) is exact; however, defining the hydrodynamic forces and moments in waves can be extremely difficult.

A high-speed craft moving in waves may transit through several regimes that have different hydrodynamic flow characteristics. For example, as the craft moves away from the crest of wave, the flow may be characterized by unsteady-state planing until the craft collides with the oncoming wave crest and enters another regime in which impact forces are important. After the impact, the craft may enter still another regime in which it is planing but in which buoyancy forces are rather significant.

The most promising approach to a method that would incorporate all three types of flow conditions into a general formulation would seem to be a modified strip theory. The mathematical justification for this approach is not rigorous; however, there is sufficient precedent to expect promising results. For example, impact loads on landing seaplanes can be estimated reasonably well using a strip theory incorporating the Wagner two-dimensional (2-D), expanding-wedge theory,[1] and Chuang[2] has provided a strip method for determining loads on an impacting prismatic form that agrees extremely well with experimental results.

More recently, Martin[3] has developed a linear strip theory for estimating motions of a planing craft at high speed, which shows good agreement with experimental results. A nonlinear model of the equations of motion would be expected to provide, in addition to the motions, reasonable estimates of the vertical accelerations which are an important consideration in designing a planing craft.

TWO-DIMENSIONAL HYDRODYNAMIC FORCE

Implicit with any strip method is the need to define the 2-D hydrodynamic force acting on an arbitrary cross section of the body. The 2-D flow problem is not simple; however, it lends itself to an empirical approach, using a combination of techniques used in hydrodynamic impact and low-aspect-ratio theories.

The typical cross section of a hard-chine, V-shaped prismatic body such as that being considered here is shown in Figure 2. Figure 2 actually illustrates two different idealized-flow conditions, assumed to represent the crossflow during unsteady planing, depending upon whether the flow separates from the chine (Figure 2a) or not (Figure 2b). Nonwetted-chine flow conditions are typical of the sections near the leading edge of the wetted length of the craft. Wetted-chine flow conditions are more typical of sections near the stern, except possibly in the most extreme motion and wave conditions. Some sections between leading edge and stern may alternate between flow conditions as the wetted length changes with the motions.

---

[*] A complete listing of references is given on page 33.

The normal hydrodynamic force per unit length f, acting at a section, is treated as quasi-steady and is assumed to contain components proportional to the rate of change of momentum and the velocity squared (drag term), i.e.

$$f = -\left\{\frac{D}{Dt}(m_a V) + C_{D,c}\rho b V^2\right\} \tag{2}$$

where V is the velocity in plane of the cross section normal to the baseline

m_a is the added mass associated with the section form

$C_{D,c}$ is the crossflow drag coefficient

ρ is the density of the fluid

b is the half beam.

For sections near the leading edge of the wetted length with nonwetted chine, the added mass is assumed to be defined in the same manner as during an impact which for a V-shaped wedge is given by

$$m_a = k_a \pi/2 \rho b^2 \tag{3}$$

where k_a is an added-mass coefficient that may also include a correction for water pileup - k_a is assumed to be 1.0 without pileup correction.

The rate of change of momentum of the fluid at a section is given by

$$\frac{D}{Dt}(m_a V) = m_a \dot{V} + V \dot{m}_a - \frac{\partial}{\partial \xi}(m_a V)\frac{d\xi}{dt} \tag{4}$$

where ξ is the body coordinate parallel to the baseline; see Figure 1. The last term on the right-hand side of Equation (4) takes into account the variation of the section added mass along the hull. This contribution can be visualized by considering the 2-D flow plane as a substantive surface moving past the body with velocity $U = -d\xi/dt$ tangent to the baseline. As the surface moves past the body, the section geometry in the moving surface may change with a resultant change in added mass. This term exists even in steady-state conditions and is the lift-producing factor in low-aspect-ratio theory.

The added mass of a section with fully wetted chines has not been developed to the same extent as the V wedge. In steady-state planing problems such as those of Shuford,[4] the crossflow is treated as a Helmholtz-type flow in which the Bobyleff results are used for estimating drag coefficients. Helmholtz flows are applicable only to steady-state conditions; so, it is assumed that the added mass for the fully wetted chine flow can be determined from Equation (3) using the value of the half-beam at the chine. In using the Shuford approach, it is assumed that the crossflow drag coefficient for a V-section is equal to the drag of a flat plate ($C_{D,c} = 1.0$) corrected by the Bobyleff flow coefficient approximated by cos β, i.e.

$$C_{D,c} = 1.0 \cos \beta \tag{5}$$

The Bobyleff flow coefficient is the theoretical ratio of the pressure on a V-section to that experienced by a flat plate for a Helmholtz-type flow.

The same approximation is used for estimating the drag coefficient for nonwetted chine sections, using the instantaneous value of the half-beam at the free surface.

An additional force acting on the body is the buoyancy force f_B. This force is assumed herein to act in the vertical direction and to be equal to the equivalent static buoyancy force multiplied by a correction factor, i.e.

$$f_B = -a\rho g(A) \tag{6}$$

where A is the cross-sectional area of the section, and a is a correction factor.
The full amount of the static buoyancy is not realized because at planing speeds the water separates from the transom and chines, reducing the pressure at these locations to atmospheric or less than the equivalent hydrostatic pressure. A greater reduction is realized in the buoyancy moment because of the corresponding shift in the center of pressure. Shuford[4] in his work on steady-state planing recommended a factor of one-half to obtain the correct buoyancy force. In the following computations, the buoyancy force was corrected by a factor of one-half, i.e., a = 1/2. The buoyancy moment, computed as the static buoyancy force multiplied by its corresponding moment arm, was corrected by an additional factor of one-half to obtain the proper mean-trim angles.

Equation (2) is a synthesis of several idealized flow conditions combined in an empirical manner. In all of these flows, it is assumed that the net relative movement of the fluid past the body is in an upward direction. This condition may not always be met in the case of unsteady planing in waves. Closer scrutiny will be required to determine what limitations will be imposed upon the problem as formulated and/or what modifications will be required to improve the formulation.

TOTAL HYDRODYNAMIC FORCE AND MOMENT

The total normal hydrodynamic force acting on the body is obtained by integrating the stripwise, 2-D, hydrodynamic force given by Equations (2) and (6) over the wetted length ℓ of the body. A body coordinate system (ξ,ζ) with its origin at CG and the ξ axis pointing forward parallel to the baseline of the body is defined in Figure 1 to facilitate this integration. The hydrodynamic force acting in the vertical or z direction of the fixed integral coordinate system is given by

$$\begin{aligned} -N\cos\theta = F_z(t) &= \int_\ell f\cos\theta\, d\xi + \int_\ell f_B\, d\xi \\ &= -\Bigg[\int_\ell \Big\{m_a(\xi,t)\dot{V}(\xi,t) + \dot{m}_a(\xi,t)V(\xi,t) \\ &\quad - U(\xi,t)\frac{\partial}{\partial\xi}\left[m_a(\xi,t)V(\xi,t)\right] \\ &\quad + C_{D,c}(\xi,t)\rho b(\xi,t)V^2(\xi,t)\Big\}\cos\theta\, d\xi \\ &\quad + a\rho g A\, d\xi\Bigg] \end{aligned} \tag{7}$$

where the integration is taken over the instantaneous wetted length. Similarly the force F_x acting in the horizontal or x direction is given by

$$\begin{aligned} F_x &= \int_\ell f\sin\theta\, d\xi \\ &= -\int \Big\{m_a(\xi,t)\dot{V}(\xi,t) + \dot{m}_a(\xi,t)V(\xi,t) \\ &\quad - U(\xi,t)\frac{\partial}{\partial\xi}\left[m_a(\xi,t)V(\xi,t)\right] \\ &\quad + C_{D,c}(\xi,t)\rho b(\xi,t)V^2(\xi,t)\Big\}\sin\theta\, d\xi \end{aligned} \tag{8}$$

Wave forces are obtained by neglecting diffraction and assuming that the wave excitation is caused both by the geometrical properties of the wave, altering the wetted length and draft of the craft, and by the vertical component of the wave orbital velocity at the surface w_z, altering the normal velocity V. The horizontal component of orbital velocity is neglected.

since it is assumed small in comparison with the forward speed $\dot{x}_{CG}$. The velocities U and V may then be written as

$$U = \dot{x}_{CG} \cos\theta - (\dot{z}_{CG} - w_z)\sin\theta$$

$$V = \dot{x}_{CG} \sin\theta - \dot{\theta}\xi + (\dot{z}_{CG} - w_z)\cos\theta \qquad (9)$$

The depth of submergence h of the body at any point $P(\xi,\zeta)$ may be determined by

$$h = z_{CG} - \xi\sin\theta + \zeta\cos\theta - r \qquad (10)$$

where r is the instantaneous value of the wave elevation directly above the point.

For regular head waves the wave elevation for a linear deepwater wave is

$$r = r_0 \cos k(x+ct) \qquad (11)$$

where r_0 is the wave amplitude

k is the wave number

c is the wave celerity.

At point $P(\xi,\zeta)$

$$x = x_{CG} + \xi\cos\theta + \zeta\sin\theta \qquad (12)$$

where $x_{CG} = \int_{\ell} \dot{x}_{CG}\, dt$

The hydrodynamic moment F_θ about CG is obtained in a similar manner by integrating over the wetted length the product of the normal force per unit length and the corresponding moment arm.

$$
\begin{aligned}
F_\theta &= -\int_\ell f(\xi,t)\xi\,d\xi - \int_\ell f_b \cos\theta\,\xi\,d\xi \\
&= \int_\ell \Big\{ m_a(\xi,t)\dot{V}(\xi,t) + \dot{m}_a(\xi,t)V(\xi,t) \\
&\quad - U(\xi,t)\frac{\partial}{\partial\xi}\left(m_a(\xi,t)V(\xi,t)\right) + C_{D,c}(\xi,t)\rho b(\xi,t)V^2(\xi,t) \\
&\quad + a\rho g A\cos\theta \Big\}\,\xi\,d\xi
\end{aligned}
\tag{13}
$$

EQUATIONS OF MOTION, GENERAL

Integrating the first term in Equations (7), (8), and (13) provides hydrodynamic forces and moments proportional to acceleration of the motion. These can be combined with the inertial terms of the rigid body to give the following equation of motion

$$
\begin{aligned}
&(M + M_a \sin^2\theta)\ddot{x}_{CG} + (M_a \sin\theta\cos\theta)\ddot{z}_{CG} - (Q_a \sin\theta)\ddot{\theta} \\
&\quad = T_x + F'_x - D\cos\theta
\end{aligned}
\tag{14}
$$

$$
\begin{aligned}
&(M_a \sin\theta\cos\theta)\ddot{x}_{CG} + (M + M_a\cos^2\theta)\ddot{z}_{CG} - (Q_a\cos\theta)\ddot{\theta} \\
&\quad = T_z + F'_z + D\sin\theta + W
\end{aligned}
$$

$$
\begin{aligned}
&-(Q_a\sin\theta)\ddot{x}_{CG} - (Q_a\cos\theta)\ddot{z}_{CG} + (I + I_a)\ddot{\theta} \\
&\quad = F'_\theta - D x_d + T x_p
\end{aligned}
$$

where $M_a(t) = \int_\ell m_a(\xi,t)\,d\xi$

$Q_a(t) = \int_\ell m_a(\xi,t)\xi\,d\xi$

$I_a(t) = \int_\ell m_a(\xi,t)\xi^2\,d\xi$

$F'_x = F_x - \left\{-(M_a\sin^2\theta)\ddot{x}_{CG} - (M_a\sin\theta\cos\theta)\ddot{z}_{CG} + (Q_a\sin\theta)\ddot{\theta}\right\}$

$F'_z = F_z - \{\text{appropriate acceleration terms}\}$

$F'_\theta = F_\theta - \{\text{appropriate acceleration terms}\}$.

A detailed evaluation of the integral expressions for the hydrodynamic forces and moments is provided in Appendix A.

The solution to Equation (14) is cumbersome; however, it can be accomplished using standard numerical techniques. Introducing the state vector $[x_1, x_2, x_3, x_4, x_5, x_6]$

where $x_1 = \dot{y}_{CG}$

$x_2 = \dot{z}_{CG}$

$x_3 = \dot{\theta}$

$x_4 = x_{CG}$

$x_5 = z_{CG}$

$x_6 = \theta$

Equation (14) can be rewritten, using matrix algebra, as

$$A\vec{\dot{x}} = \vec{g} \tag{15}$$

so that

$$\vec{\dot{x}} = A^{-1}\vec{g} \tag{16}$$

where A^{-1} is inverse of the inertial matrix A. Equation (16) is now in a form that lends itself to integration by using a numerical method such as the Runge-Kutta-Merson integration routine.

EQUATIONS OF MOTION, SIMPLIFIED FOR CONSTANT SPEED

Assuming that the perturbation velocities in the forward direction are small in comparison to the speed of the craft, the equations of motion may be further simplified by neglecting the perturbations and setting the forward velocity equal to a constant, i.e.

$$\dot{x}_{CG} = \text{CONSTANT}$$

If it is also assumed that the thrust and drag forces are small in comparison to the hydrodynamic forces and that they are acting through the center of gravity, the equations of motion may be written as

$$\ddot{x}_{CG} = 0$$

$$(M + M_a \cos^2 \theta)\ddot{z}_{CG} - (Q_a \cos \theta)\ddot{\theta} = F_z' + W$$

$$-(Q_a \cos \theta)\ddot{z}_{CG} + (I + I_a)\ddot{\theta} = F_\theta'$$

These equations also represent the case of the craft (model) being towed through CG at CONSTANT speed. Based upon the previously described equations of motion, a computer program has been written in FORTRAN language to compute the motions of a prismatic body, planing in regular head waves at high speed. A listing of the program along with the appropriate flow chart is presented in Appendix B. The listing contains reference to thrust and drag terms; however, they have no significance, except to provide a starting point for possible updating of the program to include these terms in the future.

COMPARISON OF COMPUTED RESULTS WITH EXPERIMENTS

Computations of pitch and heave motions and heave and bow accelerations were made, using the computer program for comparison with the experimental results of Fridsma.[5] Fridsma tested a series of constant-deadrise models of various lengths in regular waves to define the effects of deadrise, trim, loading, speed, length-to-beam ratio and wave proportions on the added resistance, heave and pitch motions, and impact accelerations at the bow and center of gravity. Figure 3 shows the lines of the prismatic models. The models were towed at CG with a system that permitted freedom in surge. The computer program simulates the model being towed at constant speed with CG at the baseline.

Table 1 presents some characteristics of the model and experimental conditions for which comparisons were made. Most of the comparisons have been made at a speed-to-length ratio $V/\sqrt{L}$ of 6.0 where the mathematical model is expected to be most representative. A limited comparison has also been made at $V/\sqrt{L} = 4.0$; however, no comparison has been made at $V/\sqrt{L} = 2.0$. At this speed, the model (or craft) operates in the displacement mode for which the mathematical formulation is not valid.

The average computer run corresponded to 10-second, real-time, model scale; however, only the last 2 seconds were considered free of transient effects. An example of the computer time histories of pitch and heave motions is shown in Figure 4. Although the motions are periodic, they are not perfectly sinusoidal; consequently, in determining phase relationship, the peak, positive-pitch value (bow up) and the peak, negative-heave value (maximum upward position of CG) were used as reference points. There was a difference when the opposite peaks were used.

TABLE I – MODEL CHARACTERISTICS AND WAVE CONDITIONS FOR COMPUTATIONS

(Model Length = 114.3 cm (3.75 ft); L/b = 5; C_Δ = 0.608)

CONFIGURATIONS				
SYMBOL	β deg	LCG percent L	Radius of Gyration percent L	$V/\sqrt{L}$
A	20	59.0	25.1	4.0
B	20	62.0	25.5	6.0
J	10	68.0	26.2	6.0
M	30	60.5	24.8	6.0

WAVE CONDITIONS FOR CONFIGURATION --							
A		B		J		M	
H/b	λ/L	H/b	λ/L	H/b	λ/L	H/b	λ/L
0.111	1.0	0.111	1.0	0.111	1.0	0.111	1.0
0.111	1.5	0.111	1.5	0.111	1.5	0.111	1.5
0.111	2.0	0.111	2.0	0.111	2.0	0.111	2.0
0.111	3.0	0.111	3.0	0.111	3.0	0.111	3.0
0.111	4.0	0.111	4.0	0.111	4.0	0.111	4.0
0.111	6.0	0.222	6.0	0.111	6.0	0.111	6.0
		0.334	4.0				
		0.111	6.0				

Corresponding time histories of bow and CG accelerations are shown in Figure 5. The bow acceleration was computed at Station 0. As can be seen in these plots, the impact accelerations ranged in magnitude from cycle to cycle. The maximum impact (or negative value) acceleration computed during the final 2 seconds of run was used in the comparisons with experimental values. In some instances, particularly near resonance, the maximum impact acceleration was more than twice the average impact value.

Figure 6 shows a comparison of variation of computed and experimental pitch and heave motion with wave height for the 20-degree deadrise model in a 15-foot wavelength and for a speed-to-length ratio of 6.0. Figure 7 shows the corresponding impact acceleration at the bow and CG. The computed results closely follow the experimental data, except for CG acceleration at the extreme wave height condition, where the computed value is apparently much lower. Experimental data show that the model was leaving the water at this wave-height condition. The computer model did not leave the water but came very close;

see Figure 8. Figure 8 is a trajectory of the computer model relative to the wave for a selected cycle of motion. The computer model behaves very much as expected. On the left-hand side of the figure, the craft is planing down the crest of the wave and, as it approaches the wave trough, comes very close to leaving the water before slamming and submerging itself deeply into the front of the oncoming wave crest.

Figures 9 through 14 show comparisons of the computed and experimental pitch and heave motions at $V/\sqrt{L} = 6.0$ through a range of wavelengths and at a constant wave height of 2.54 centimeters (1 inch) for deadrise models with 10, 20, and 30 degrees. The data have been plotted with respect to the coefficient C_λ, defined by Fridsma as $L/\lambda\,[C_\Delta/(L/2b)^2]^{1/3}$. Note that in our notation, b is the half-beam.

Comparisons of heave and pitch for the 10-degree deadrise model shown in Figures 9 and 10, respectively, show excellent results. The computer model accurately predicts the secondary peaks in the pitch and heave responses at $C_\lambda = 0.19$. At this condition, the physical experimental model rebounds so as to fly over alternate waves. The computer model oscillates at half the wave-encounter frequency and comes close to leaving the water at alternate encounters with the wave. It does not quite leave the water to fly over alternate wave crests; nonetheless, it is a good representation of the actual motion.

The heave and pitch comparison for the 20-degree deadrise model at $V/\sqrt{L} = 6.0$ is also excellent as can be seen in Figures 11 and 12, respectively. No experimental phase data for the condition were reported for C_λ greater than 0.072; however, extrapolated results (not shown) are in line with the computed results. The pitch and heave results shown in Figures 13 and 14 for the 30-degree deadrise model are good; however, responses at $C_\lambda = 0.048$ and $C_\lambda = 0.072$ are higher than the experimental results.

For practical considerations a computational scheme for planing boat motions should be valid for a range from approximately $V/\sqrt{L} = 4.0$ to $V/\sqrt{L} = 6.0$. Computations of the motions were made for $V/\sqrt{L} = 4.0$ for the 20-degree deadrise model; see Figures 15 and 16. Again the comparison of the computed heave and pitch response with experimental results is excellent.

Comparisons of the computed and experimental impact accelerations (or largest negative values) are presented in Figures 17 through 20. Figures 17 and 18 show bow and CG accelerations for the 10-degree deadrise model; Figure 19 shows similar results for the 20-degree deadrise model. Figure 20 shows the results for the 30-degree deadrise model. In all cases, the comparison appears to be fair to good. In the shorter wavelengths, $\lambda/L = 1.0$ and $\lambda/L = 1.5$, the computed accelerations are higher than the corresponding experimental values. This is most pronounced for the 10-degree deadrise angle model.

CONCLUSIONS AND RECOMMENDATIONS

A mathematical model of a craft having a constant deadrise angle, planing in regular waves, has been formulated using a modified low-aspect-ratio or strip theory. It was assumed that the wavelengths were long in comparison to the craft length and that the wave slopes were small. The coefficients in the equations of motion were determined by a combination of theoretical and empirical relationships.

A simplified version for the case of a craft or model being towed at constant speed was programed for computations on a digital computer, and the results were compared with existing experimental data.

The comparison of the computed pitch and heave motions and phase angles with the corresponding experimental data gave remarkably satisfying results. Comparison of the bow and CG accelerations was fair to good.

In summary, the previously described mathematical model appears to be a valid representation of a planing craft in waves for the specific craft geometry and wave conditions considered.

To make the computer program more valuable to the designer the following additional work is recommended:

1. Improve estimates of hydrodynamic coefficients to obtain better acceleration data and to include more complicated ship geometry.
2. Determine added resistance in waves.
3. Include freedom to surge and to add components of propulsion.
4. Extend to the case of irregular waves.

ACKNOWLEDGMENTS

Acknowledgment is given to Dr. Joseph Whalen and Ms. Sue Fowler of Operations Research, Inc., who translated the equations of motion into an operational computer program.

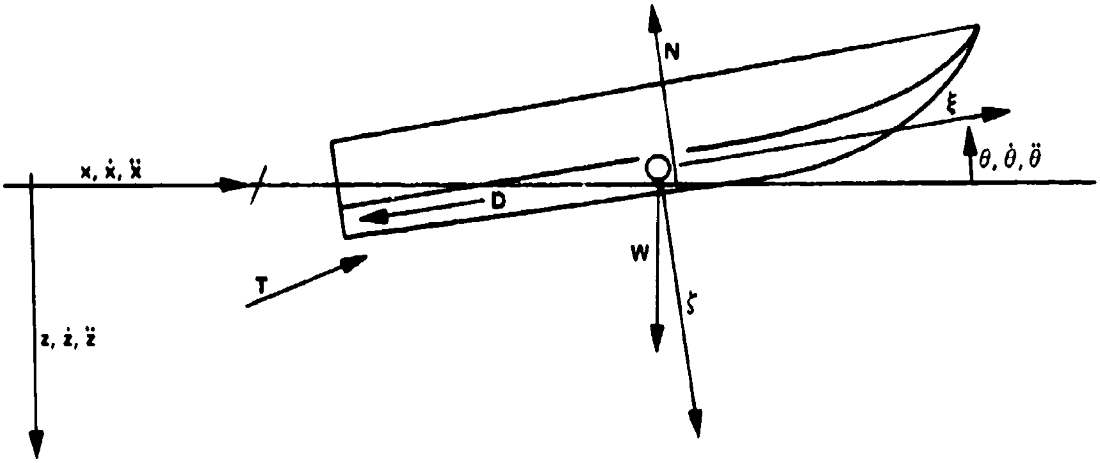


Figure 1 – Coordinate System

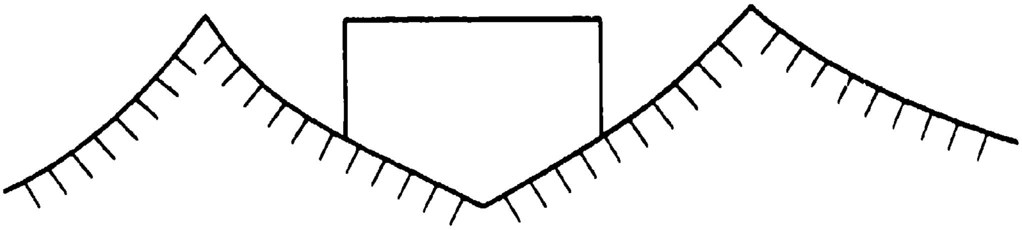


Figure 2a – Flow Separation from Chine

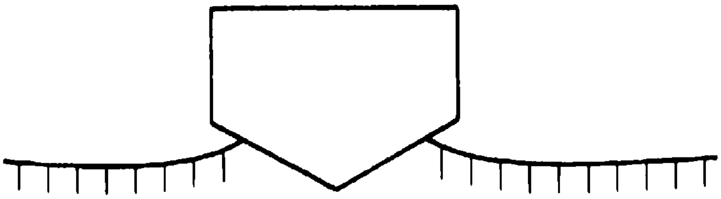


Figure 2b – Nonwetted Chine

Figure 2 – Types of Two-Dimensional Flow

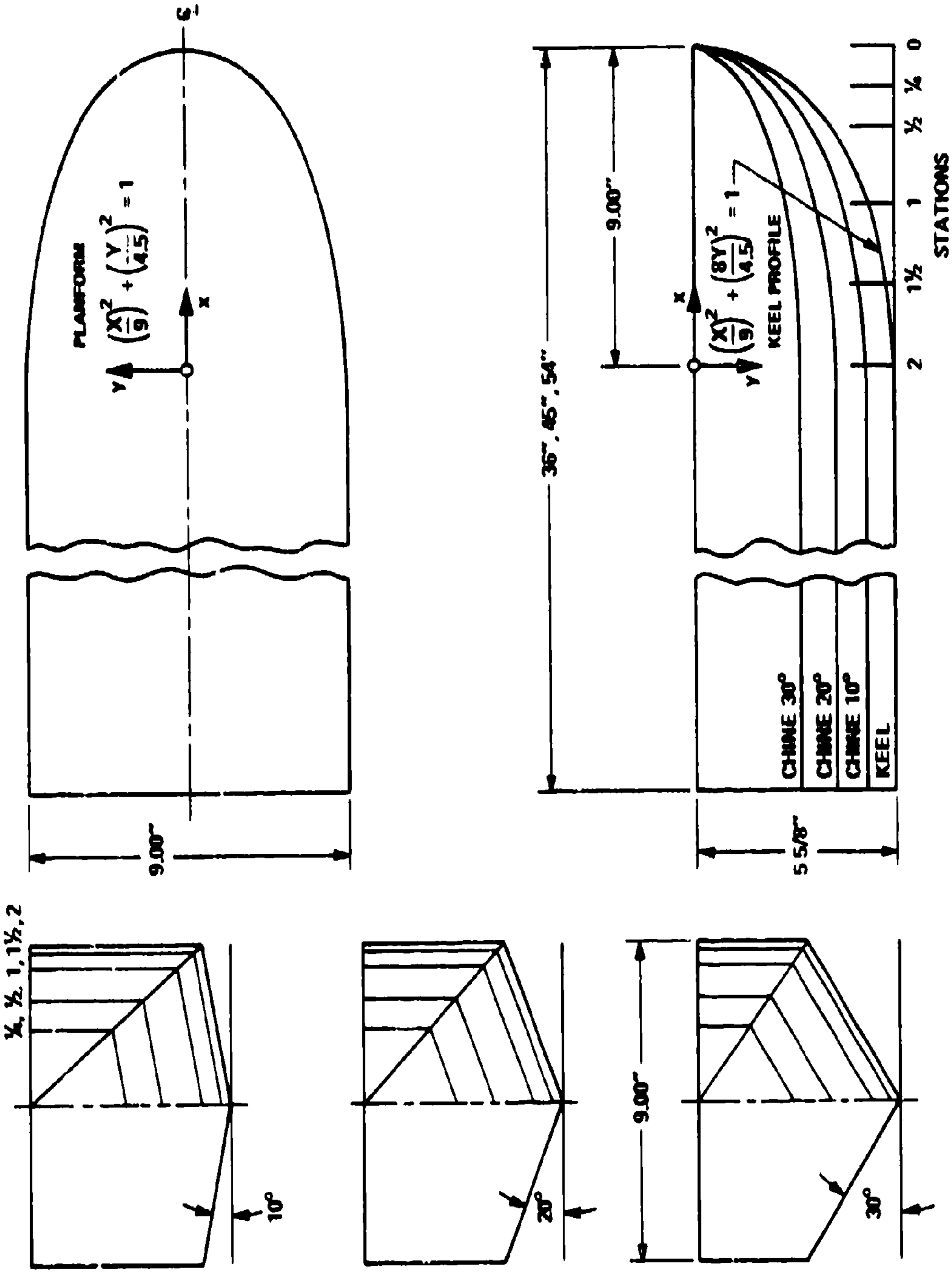


Figure 3 – Lines of Prismatic Models
(From Reference 5)

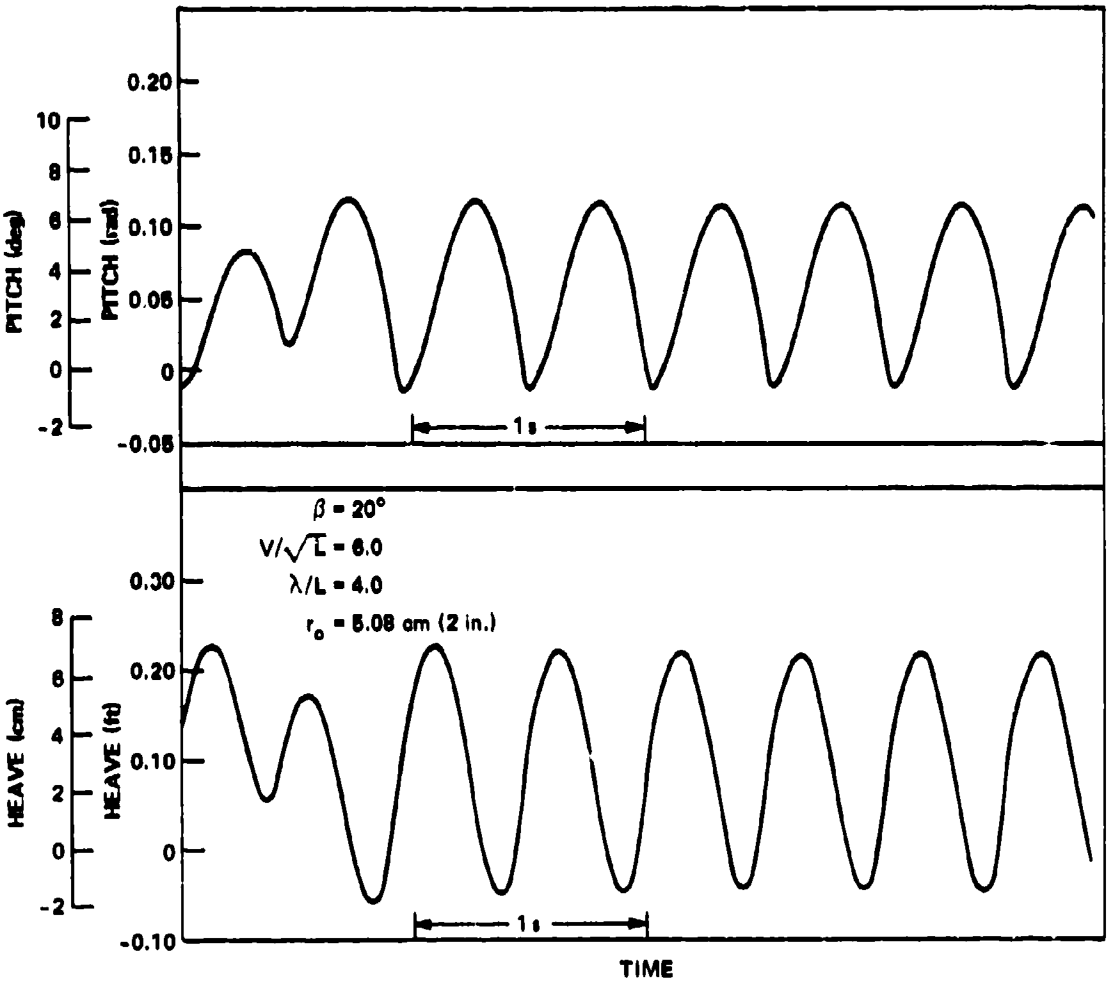


Figure 4 – Sample Time Histories of Computed Pitch and Heave Motions

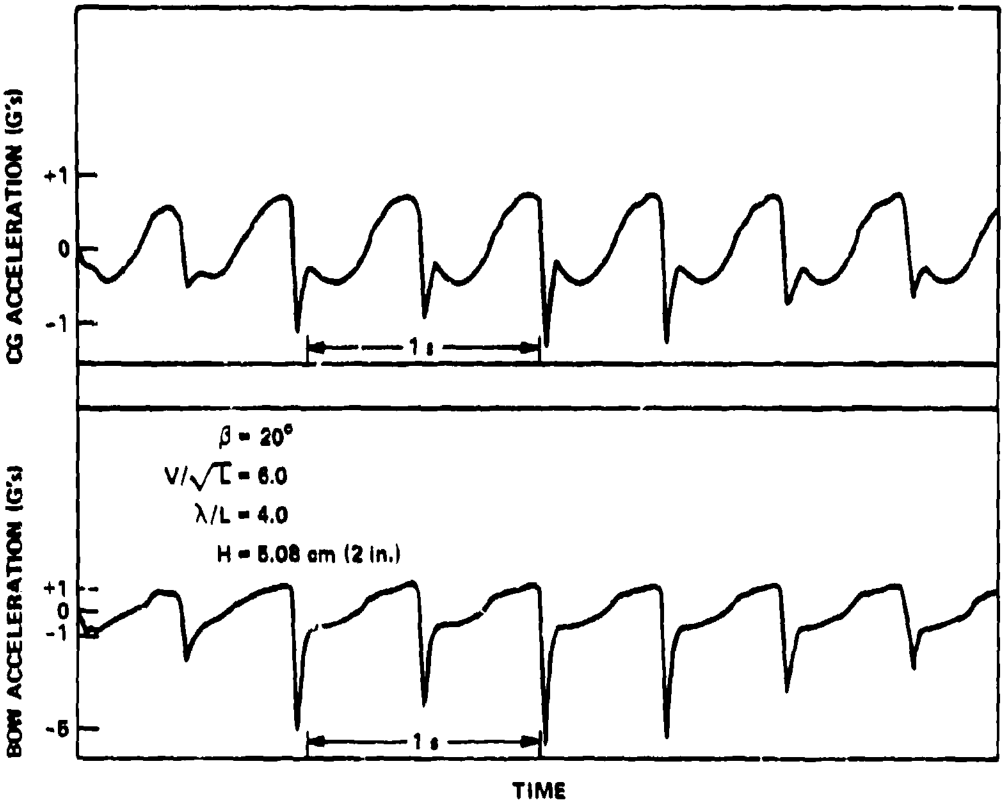


Figure 5 – Sample Time Histories of Computed Accelerations of Bow and Center of Gravity

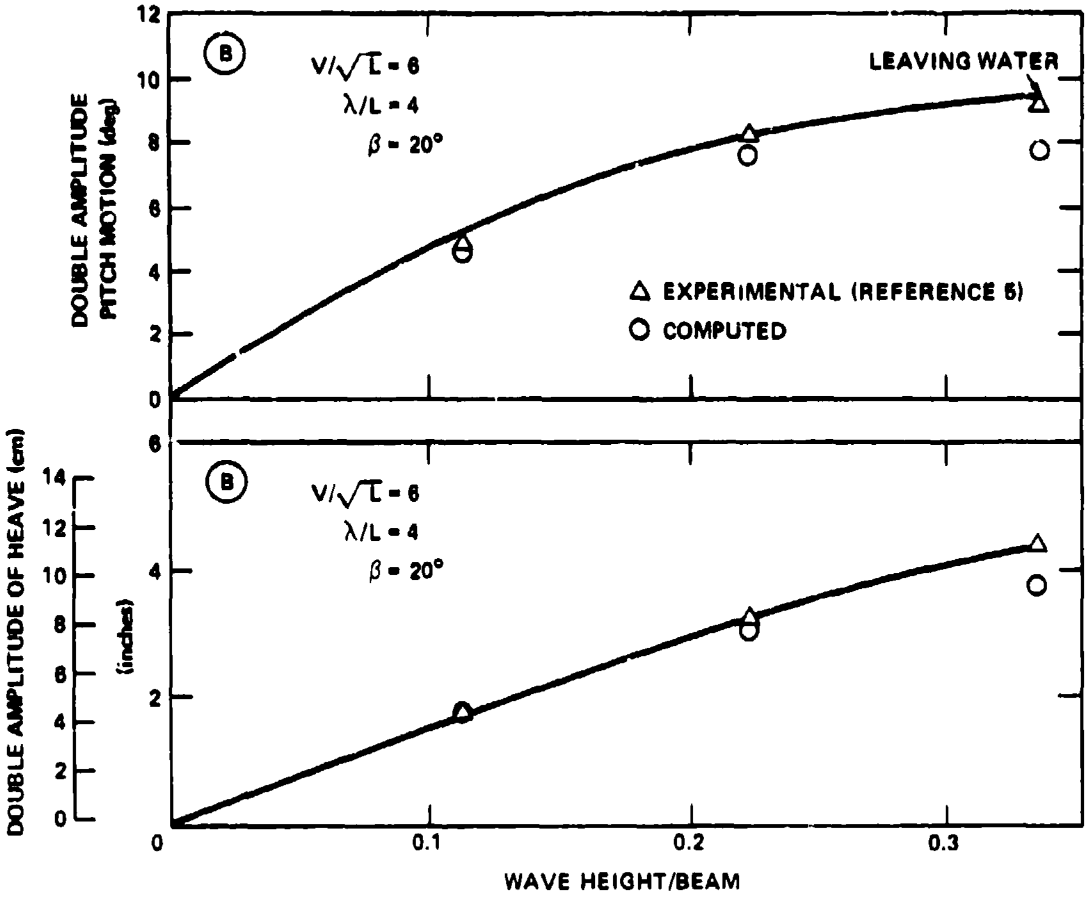


Figure 6 – Variation of Pitch and Heave with Wave Height

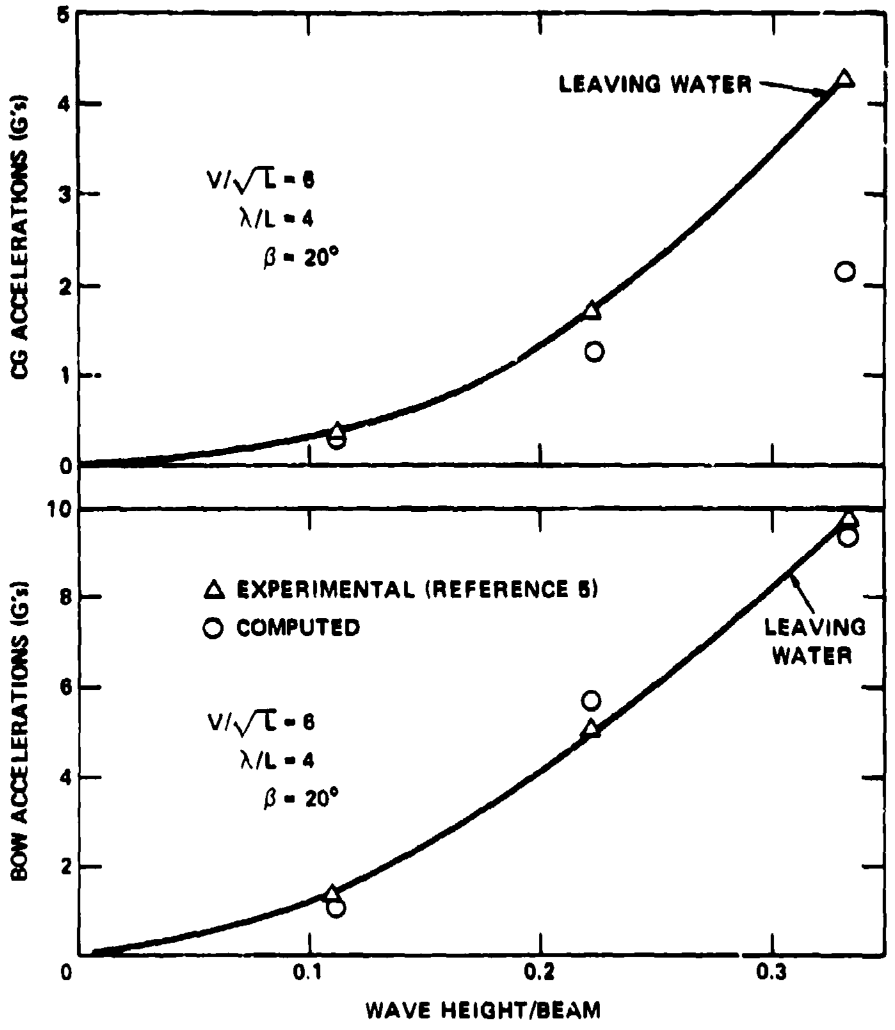


Figure 7 – Variation of Acceleration of Bow and Center of Gravity with Wave Height

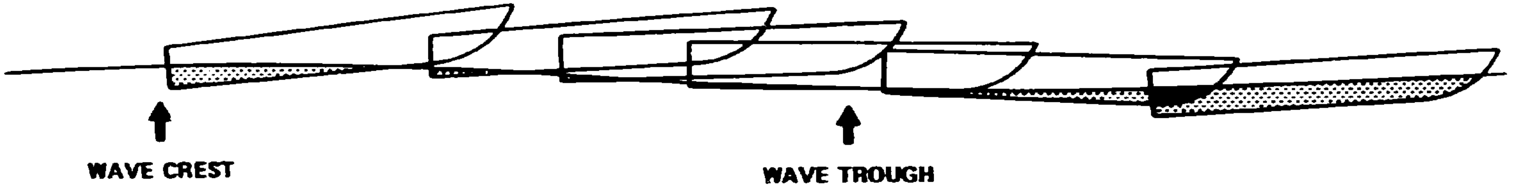


Figure 8 – Trajectory of Computer Model Relative to Wave

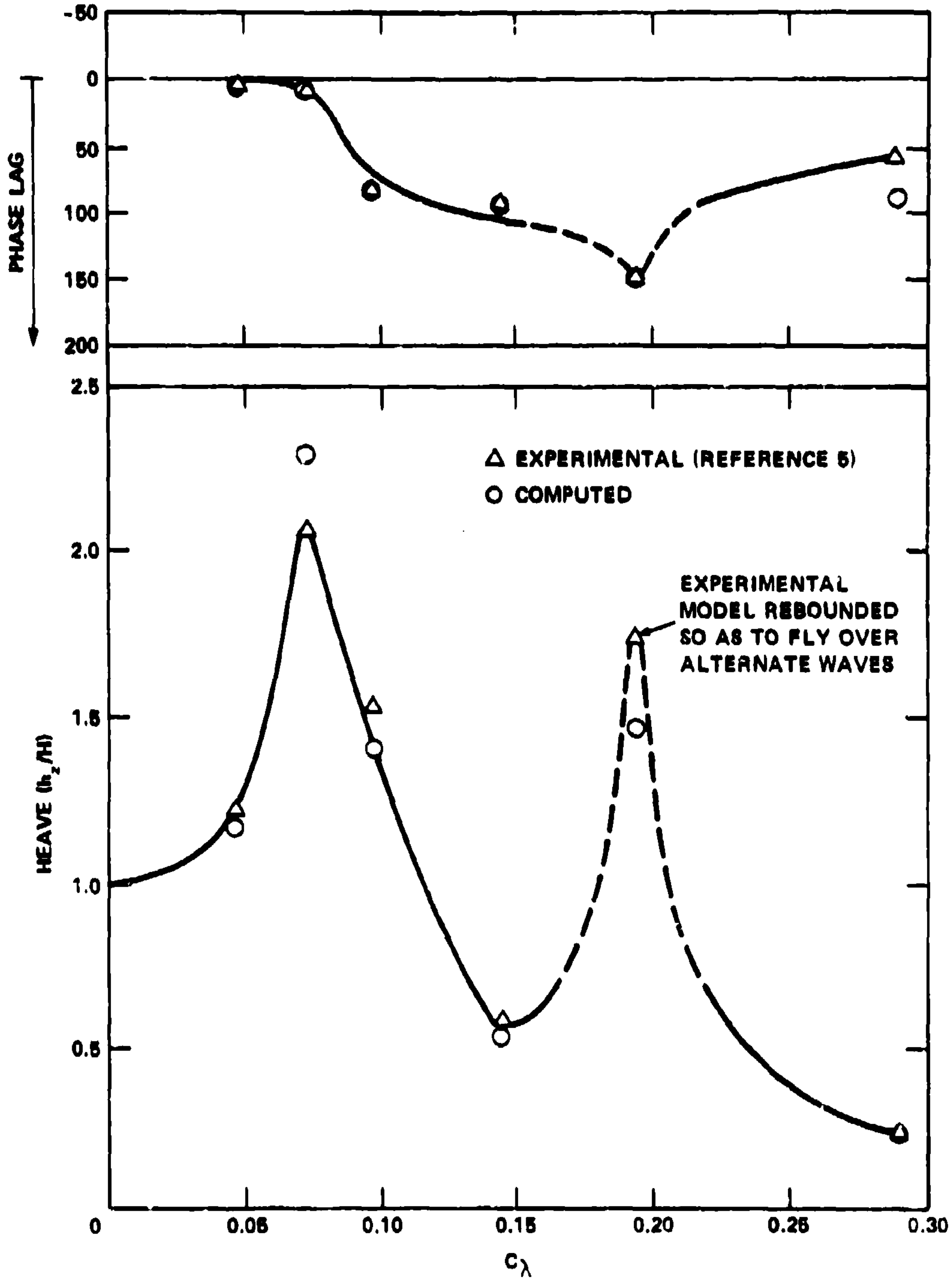


Figure 9 – Heave Response for 10-Degree Deadrise Model at $V/\sqrt{L} = 6.0$

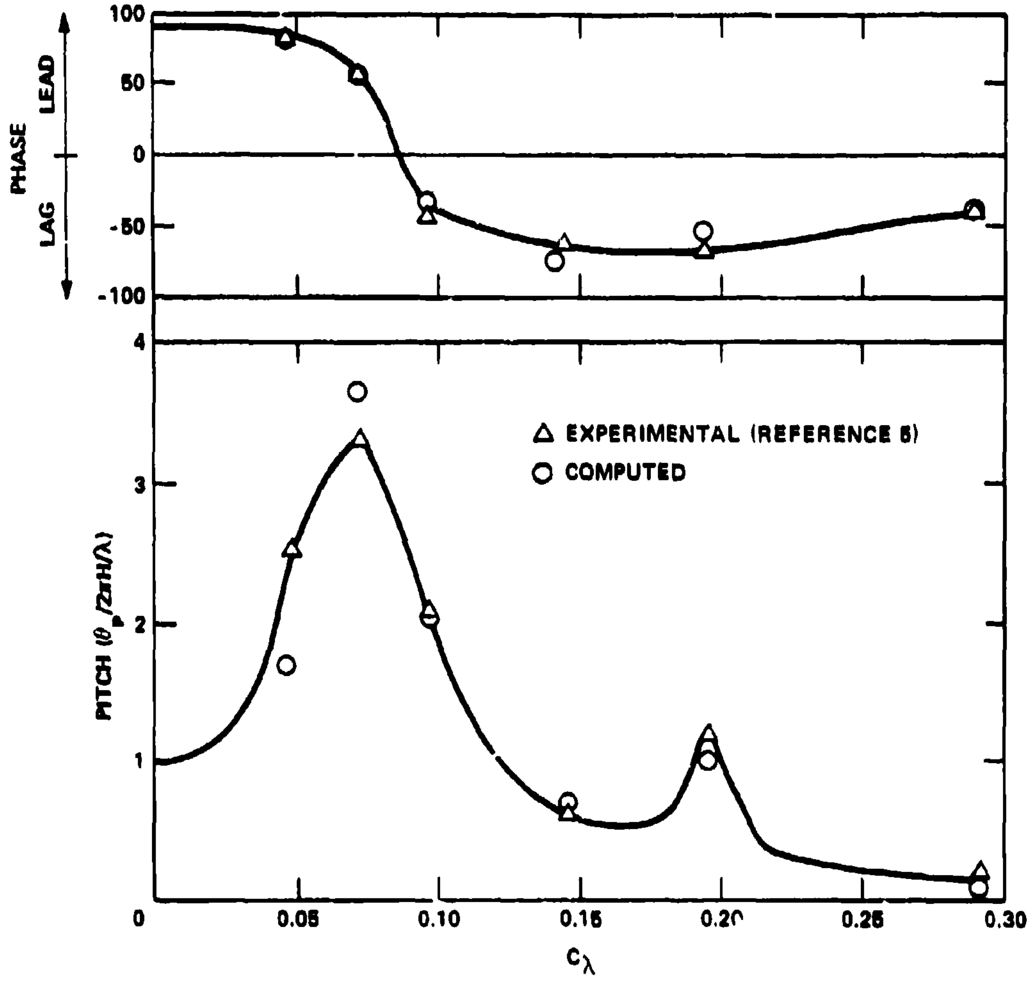


Figure 10 – Pitch Response for 10-Degree Deadrise Model at $V/\sqrt{L} = 6.0$

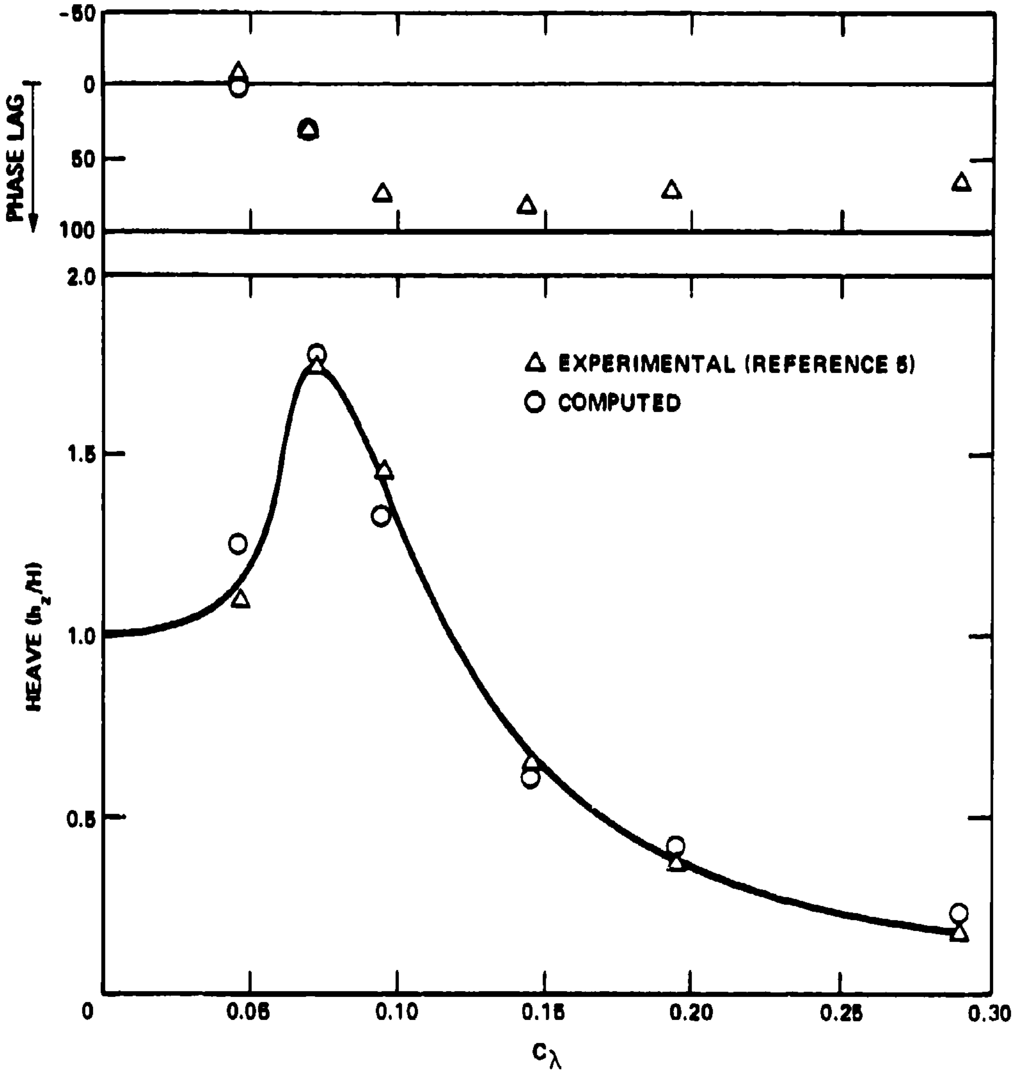


Figure 11 – Heave Response for 20-Degree Deadrise Model at $V/\sqrt{L} = 6.0$

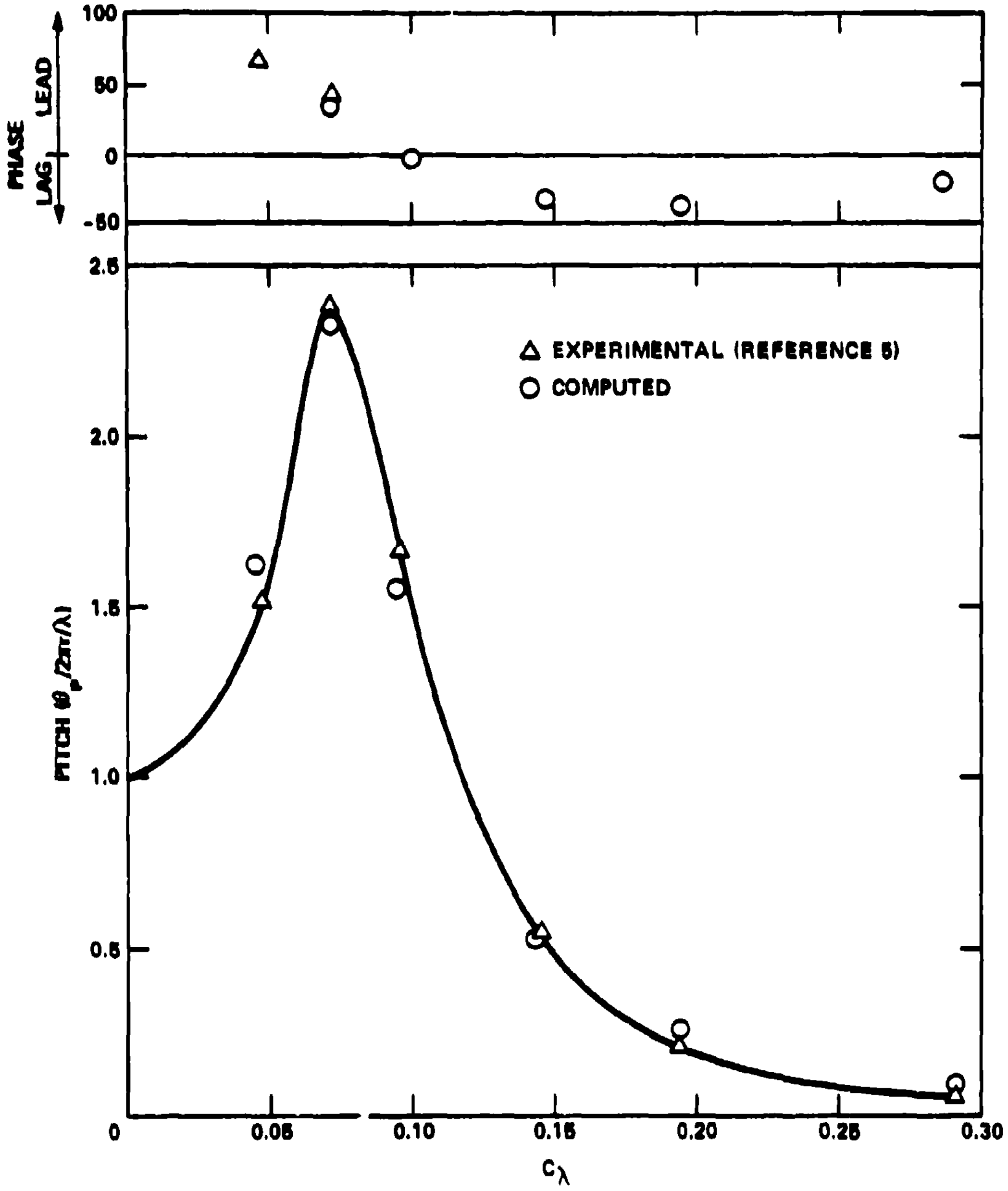


Figure 12 – Pitch Response for 20-Degree Deadrise Model at $V/\sqrt{L} = 6.0$

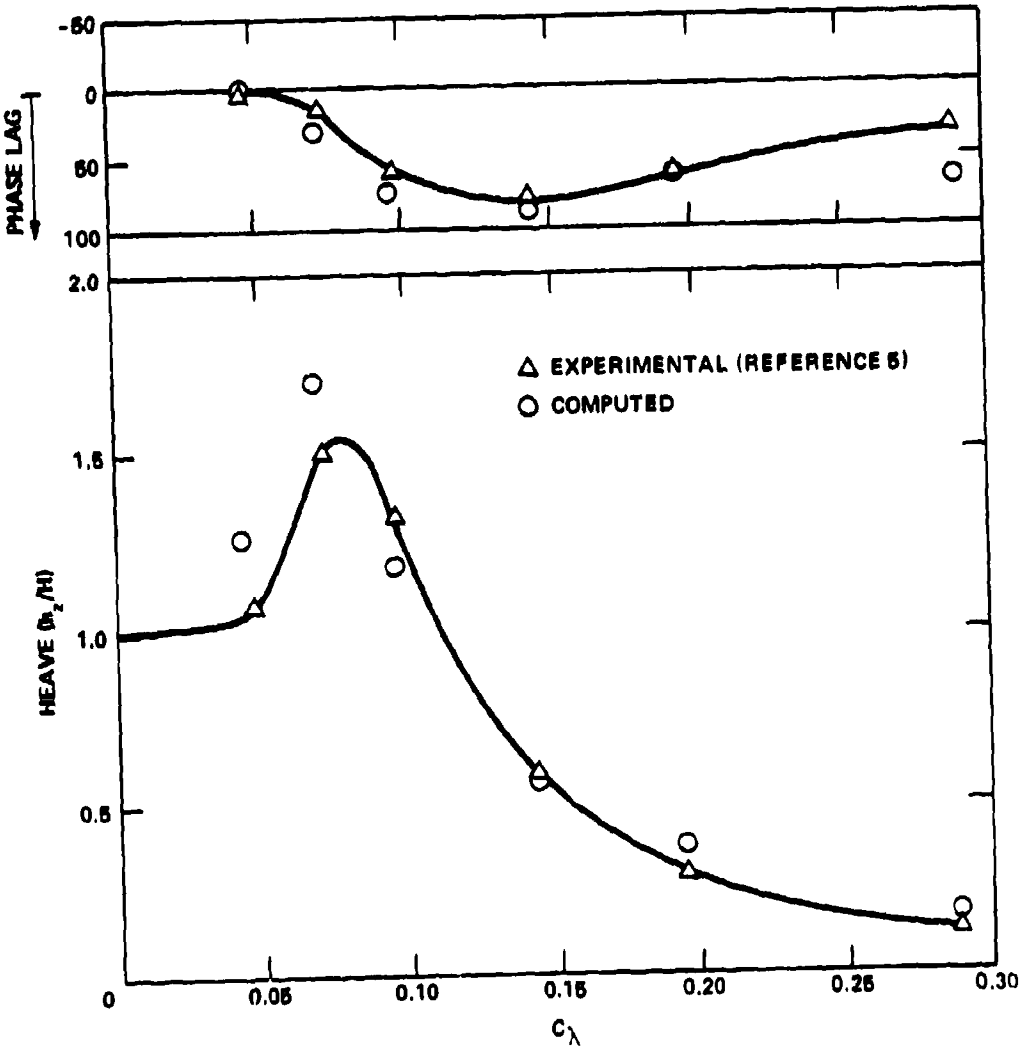


Figure 13 – Heave Response for 30-Degree Deadrise Model at $V/\sqrt{L} = 6.0$

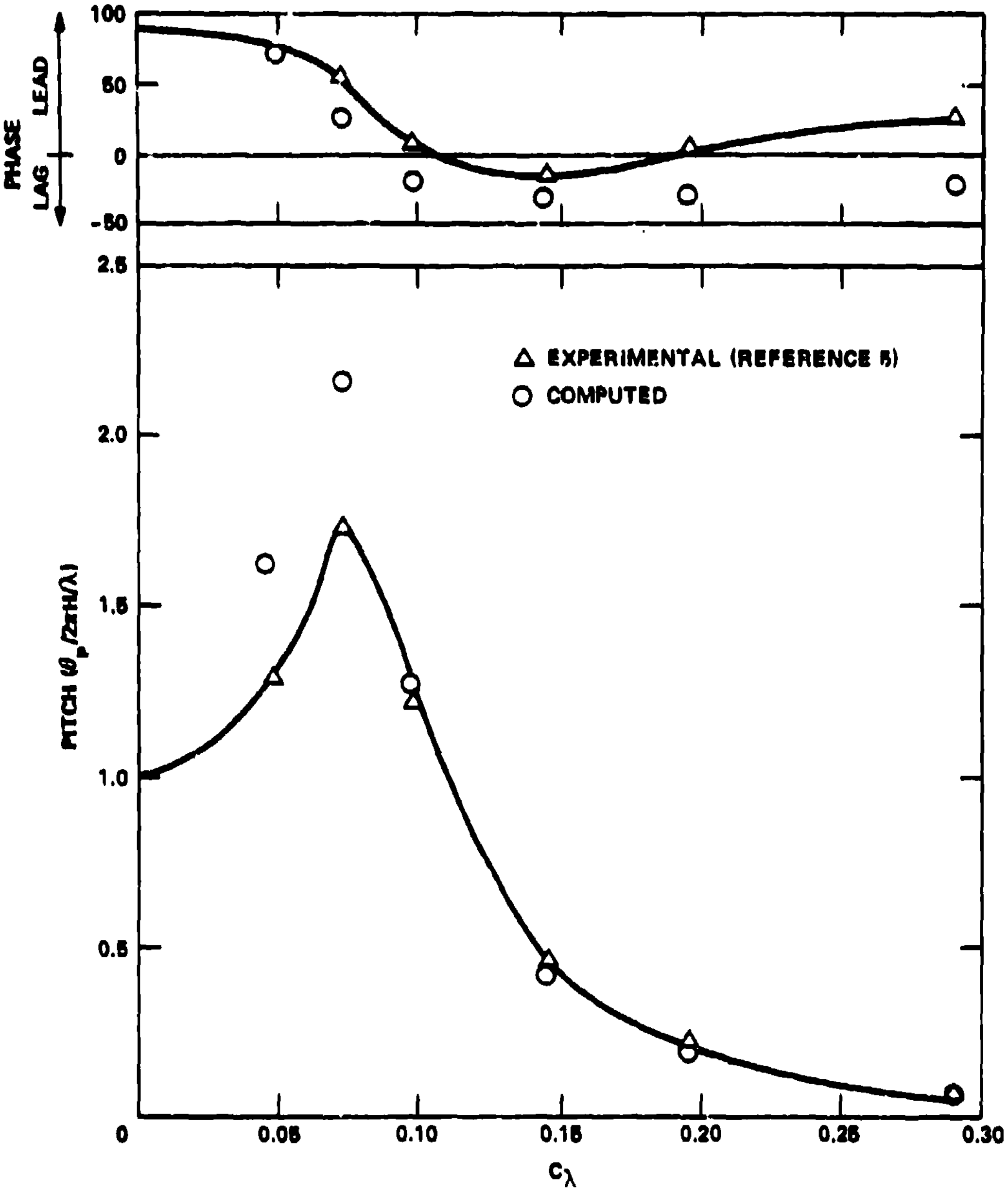


Figure 14 – Pitch Response for 30-Degree Deadrise Model at $V/\sqrt{L} = 6.0$

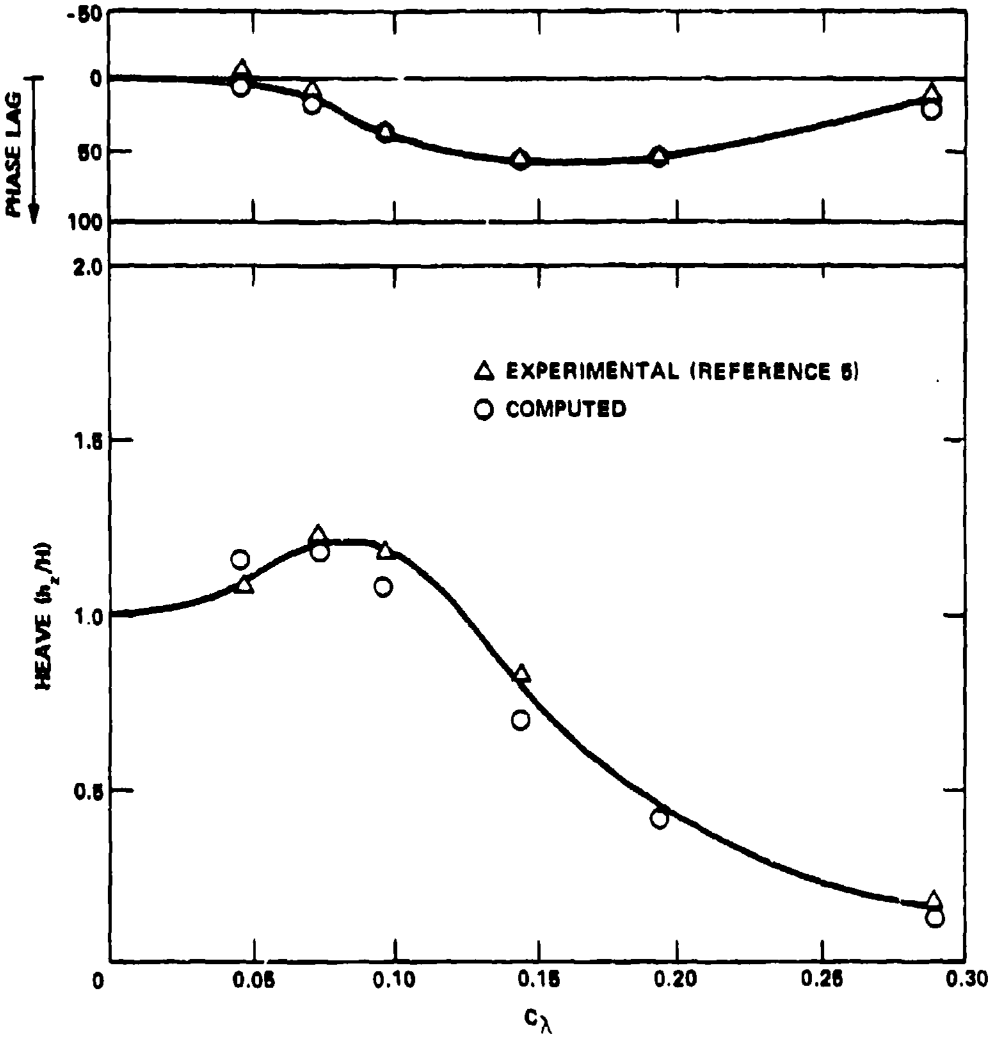


Figure 15 – Heave Response for 20-Degree Deadrise Model at $V/\sqrt{L} = 4.0$

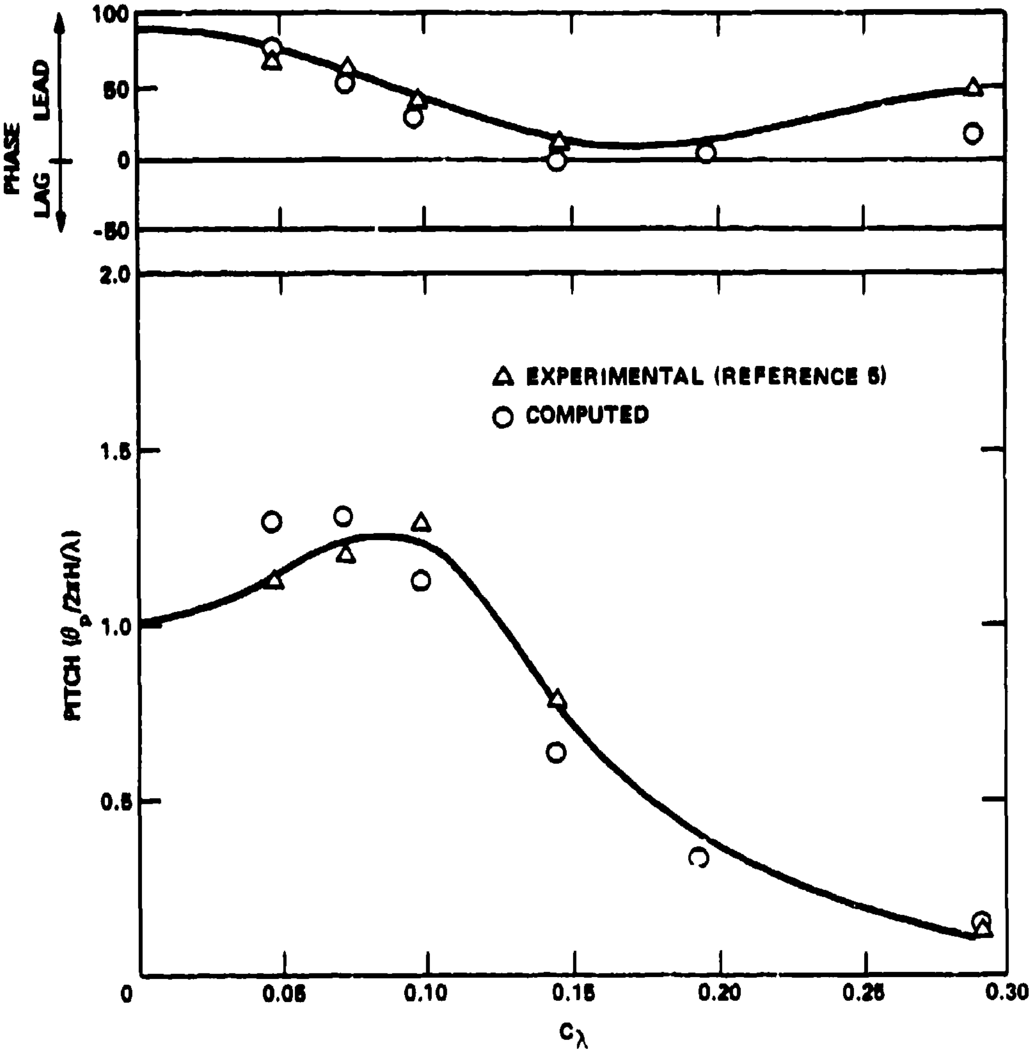


Figure 16 – Pitch Response for 20-Degree Deadrise Model at $V/\sqrt{L} = 4.0$

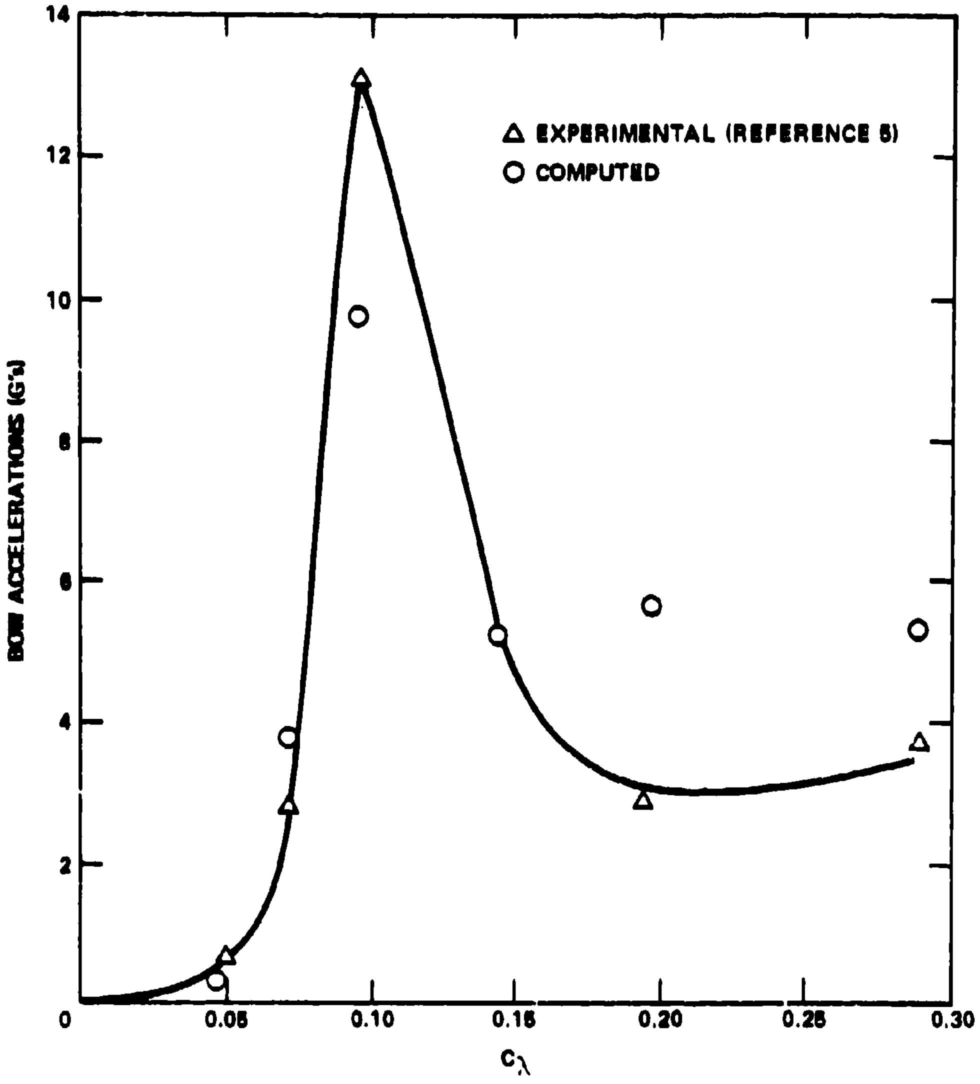


Figure 17 – Bow Acceleration for 10-Degree Deadrise Model at $V/\sqrt{L} = 6.0$

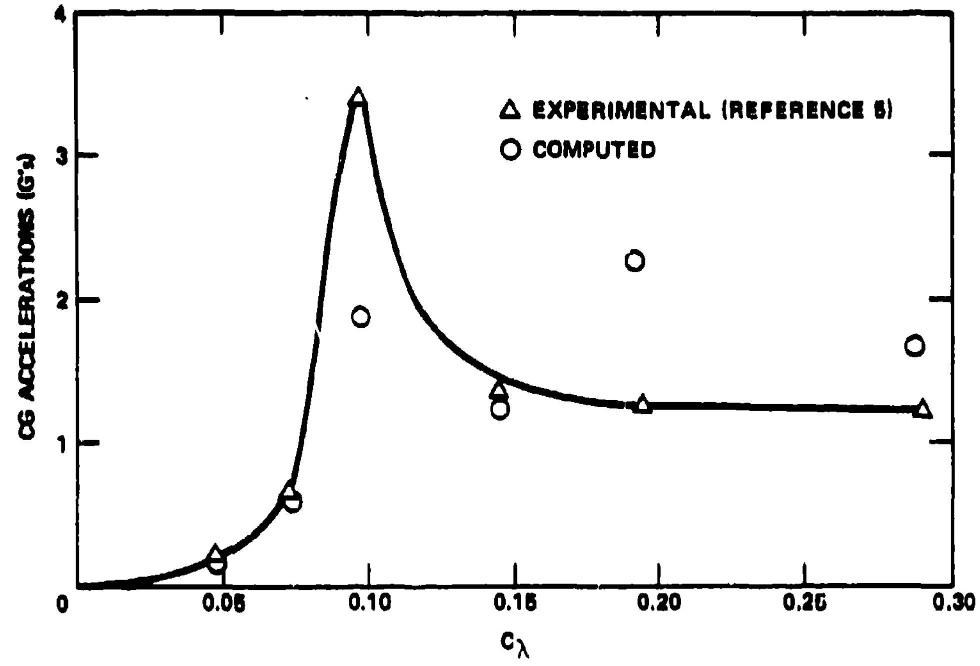


Figure 18 – Center of Gravity Acceleration for 10-Degree Deadrise Model at $V/\sqrt{L} = 6.0$

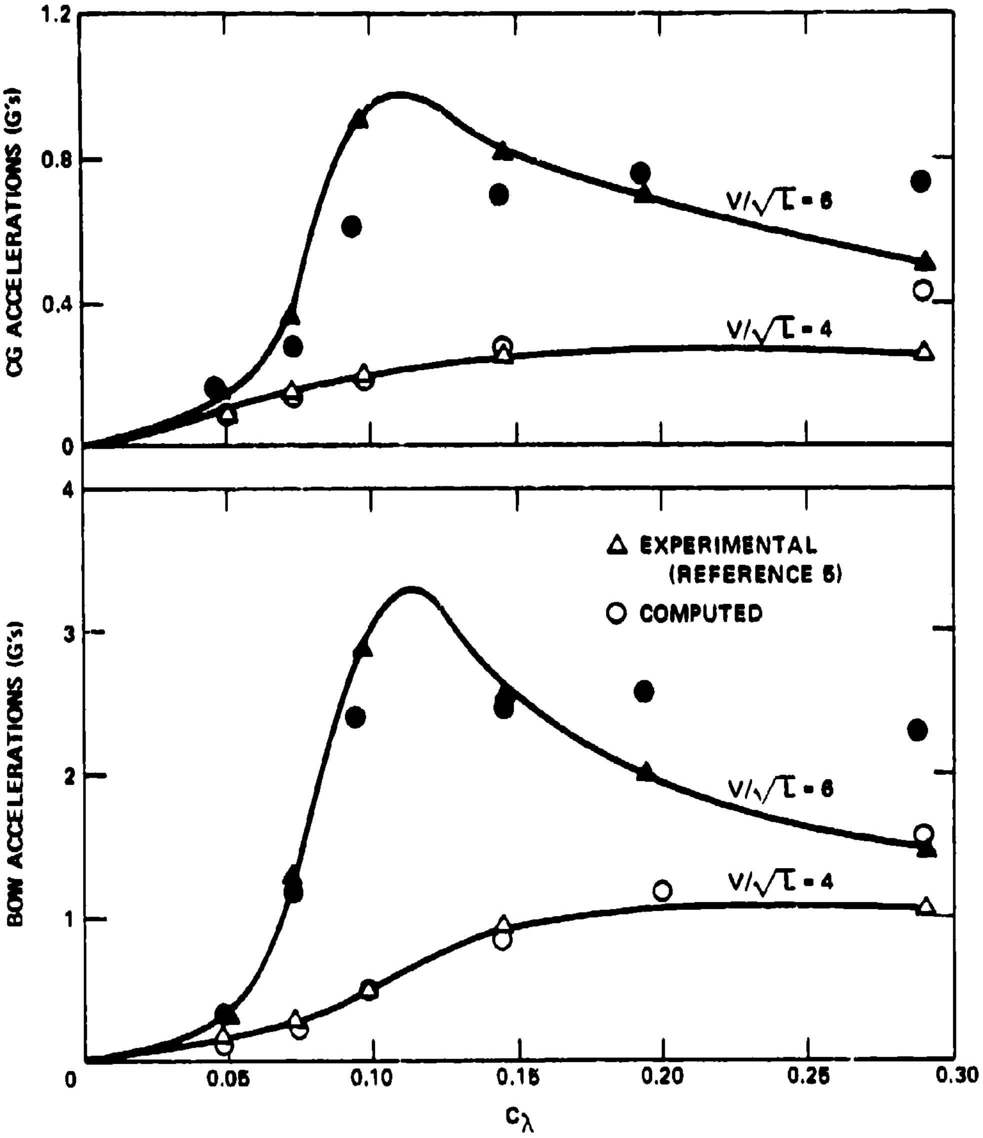


Figure 19 – Bow and Center of Gravity Accelerations for 20-Degree Deadrise Model at $V/\sqrt{L} = 4.0$ and $V/\sqrt{L} = 6.0$

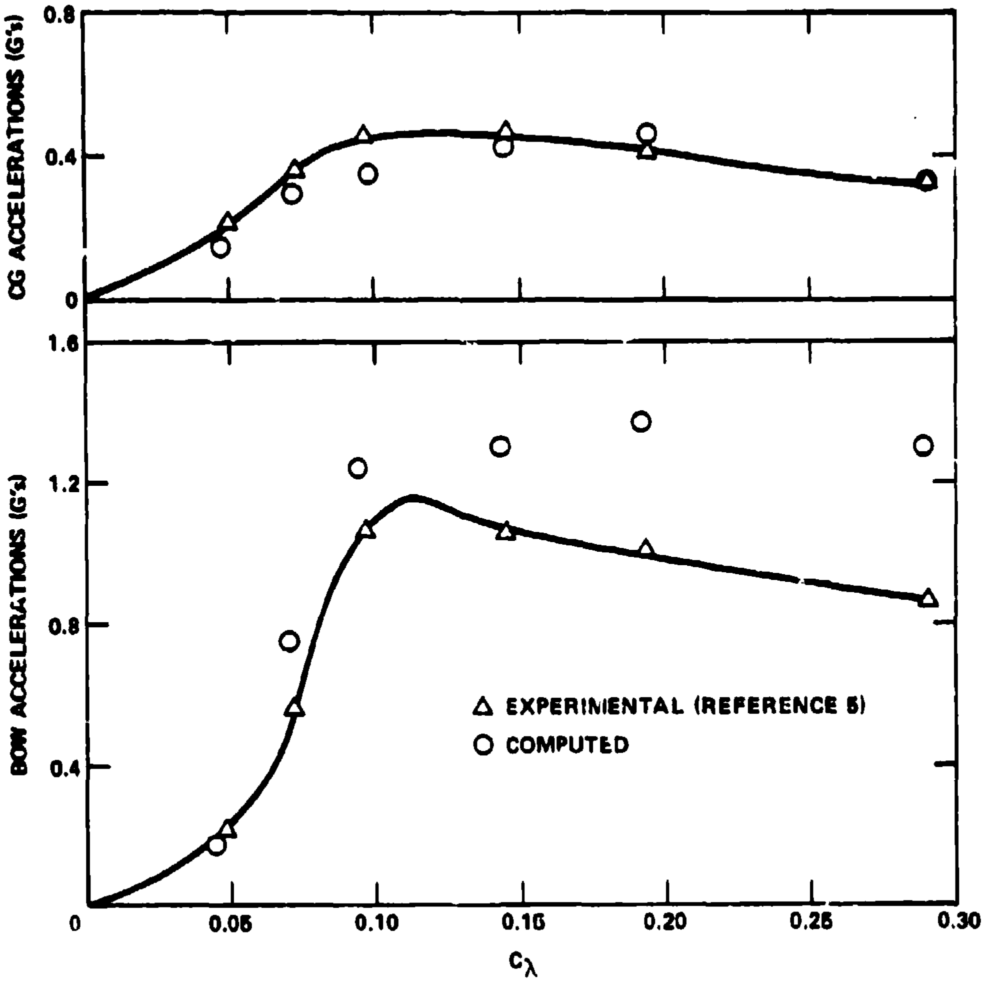


Figure 20 – Bow and Center of Gravity Accelerations for 30-Degree Deadrise Model at $V/\sqrt{L} = 6.0$

REFERENCES

1. Wagner, H., *"Landing of Seaplanes,"* leitschrift fur Flegtechnik und Motorluflsshiff-fahrt, (14 Jan 1931); National Advisory Committee for Aeronautics TM 672 (May 1931).

2. Chuang, S.L., *"Slamming Tests of Three-Dimensional Models in Calm Water and Waves,"* NSRDC Report 4095 (Sep 1973).

3. Martin, M., *"Theoretical Predictions of Motions of High-Speed Planing Boats in Waves,"* DTNSRDC Report 76-0069 (Apr 1976).

4. Shuford, S.L., Jr., *"A Theoretical and Experimental Study of Planing Surfaces Including Effects of Cross Section and Plan Form,"* National Advisory Committee for Aeronautics Report 1355 (1957).

5. Fridsma, G., *"A Systematic Study of the Rough-Water Performance of Planing Boats,"* Davidson Laboratory, Stevens Institute of Technology Report R1275 (Nov 1969).

APPENDIX A
EVALUATION OF HYDRODYNAMIC FORCE AND MOMENT INTEGRALS

The hydrodynamic force the craft experiences in the vertical direction as derived in the text is:

$$F_z = -\int_{\ell} \left\{ m_a \dot{V} - U \frac{\partial m_a V}{\partial \xi} + \dot{m}_a V + C_D \rho b V^2 \right\} \cos\theta \, d\xi + \int_{\ell} a \rho g A \, d\xi$$

where $U = \dot{x}_{CG} \cos\theta - (\dot{z} - w_z) \sin\theta$

and

$$V = \dot{x}_{CG} \sin\theta + (\dot{z} - w_z) \cos\theta - \dot{\theta}\xi$$

Another force acting in the vertical direction is the weight of the craft.

The first two terms of the integral are evaluated by making the substitutions

$$\begin{aligned}
\dot{V} &= \ddot{x}_{CG} \sin\theta - \ddot{\theta}\xi + \ddot{z}_{CG} \cos\theta - \dot{w}_z \cos\theta \\
&\quad + \dot{\theta}(\dot{x}_{CG} \cos\theta - \dot{z}_{CG} \sin\theta) + w_z \dot{\theta} \sin\theta \\
\frac{\partial V}{\partial \xi} &= -\dot{\theta} - \frac{\partial w_z}{\partial \xi} \cos\theta \\
\frac{\partial U}{\partial \xi} &= \frac{\partial w_z}{\partial \xi} \sin\theta \\
\frac{d w_z}{dt} &= \dot{w}_z - U \frac{\partial w_z}{\partial \xi}
\end{aligned}$$

and noting that

$$\int_{\ell} UV \frac{\partial m_a}{\partial \xi} d\xi = -UVm_a \Big|_{stern} - \int_{\ell} m_a \frac{\partial UV}{\partial \xi} d\xi$$

Using the previously described substitutions, the force becomes

Preceding Page Blank -

$$
\begin{aligned}
F_z = \Big\{ & -M_a \cos\theta\, \ddot{z}_{CG} - M_a \sin\theta\, \ddot{x}_{CG} + Q_a \ddot{\theta} + M_a \dot{\theta} (\dot{z}_{CG} \sin\theta - \dot{x}_{CG} \cos\theta) \\
& + \int_{\ell} m_a \frac{d w_z}{dt} \cos\theta \, d\xi - \int_{\ell} m_a w_z \dot{\theta} \sin\theta \, d\xi \\
& - \int_{\ell} m_a V \frac{\partial w_z}{\partial \xi} \sin\theta \, d\xi + \int_{\ell} m_a U \frac{\partial w_z}{\partial \xi} \cos\theta \, d\xi \\
& - UVm_a \Big|_{stern} - \int_{\ell} V \dot{m}_a \, d\xi - \rho \int_{\ell} C_{D,c}\, b V^2 d\xi \Big\} \cos\theta \\
& + \int_{\ell} a \rho g A \, d\zeta
\end{aligned}
$$

where $M_a = \int_{\ell} m_a \, d\xi$

and

$$Q_a = \int_{\ell} m_a \xi \, d\xi$$

This is essentially the form in which the integrals have been computed in the program.

The rate of change of the sectional added mass in the third term of the integral expression is derived by relating it to the rate of change of depth of fluid penetration of the section. The added mass of a section is assumed to be equal to

$$m_a = k_a \, \pi/2 \, \rho b^2$$

for which the time derivative is

$$\dot{m}_a = k_a \pi \rho b \dot{b}$$

where b is the instantaneous half-beam of the section, and k_a is an added-mass coefficient, assumed to be constant. A value of $k_a = 1.0$ was used in the computations contained in this report. For sections with constant deadrise, which is an imposed limitation of this work, the half-beam is related to the depth of penetration by

$$b = d \cot \beta$$

where d is depth of penetration, and β is deadrise angle.

Taking into account the effect of water pileup, the effective depth of penetration d_e is, according to Wagner

$$d_e = \pi/2\ d$$

and

$$b = d_e \cot\beta = \pi/2\ d \cot\beta$$

where $\pi/2$ is the factor by which the wedge immersion is increased by the pileup. Using this expression for the half-beam, the rate of change of sectional added mass becomes

$$\dot{m}_a = k a \pi \rho b (\pi/2 \cot\beta) \dot{d}$$

This expression is valid for penetration of the section up to the chine. When the immersion exceeds the chine, the sectional added mass is assumed to be constant, i.e.,

$$m_a = k\ \pi/2\ \rho b_{max}^2$$
$$\dot{m}_a = 0$$

where b_{max} is the half-beam at chine.

The submergence of a section in terms of the motions is given by

$$h = z - r$$

where $z = z_{CG} - \xi \sin\theta + \zeta \cos\theta$

$$r = r_0 \cos\left\{\kappa(x_{CG} + \xi\cos\theta + \zeta\sin\theta) + \omega t\right\}$$

For wavelengths which are long in comparison to the draft and for small wave slopes, the immersion of a section measured perpendicular to the baseline is approximately

$$d \approx \frac{z - r}{\cos\theta - \upsilon \sin\theta}$$

where υ = wave slope

The rate change of submergence d is given by

$$\dot{d} = \frac{\dot{z} - \dot{r}}{\cos\theta - \upsilon\sin\theta} + \frac{(z - r)}{(\cos\theta - \upsilon\sin\theta)^2} \cdot \frac{\partial(\cos\theta - \upsilon\sin\theta)}{\partial t}$$

Since immersion (z - r) is always small in the valid range of the previously described expression, the relationship can be further simplified to

$$\dot{d} \approx \frac{\dot{z} - \dot{r}}{\cos\theta - \upsilon\sin\theta}$$

and

$$\dot{m}_a \approx k_a \pi \rho b(\pi/2 \cot\beta) \frac{(\dot{z} - \dot{r})}{\cos\theta - \upsilon\sin\theta}$$

The expansion of the integral expression for the hydrodynamic moment in pitch follows the procedure used for the vertical force. The results are summarized as follows

$$\begin{aligned}
F_\theta = & -I_a\ddot{\theta} + Q_a\cos\theta\,\ddot{z}_{CG} - Q_a\dot{\theta}(\dot{z}_{CG}\sin\theta - \dot{x}_{CG}\cos\theta) \\
& - \int_{\ell} m_a \cos\theta \frac{dw_z}{dt} \xi d\xi + \int_{\ell} m_a \dot{\theta} \sin\theta\, w_z \xi d\xi \\
& + \int_{\ell} V \dot{m}_a \xi d\xi + \int_{\ell} \rho C_D b V^2 \xi d\xi \\
& + m_a U V \xi \Big|_{\text{stern}} + \int_{\ell} m_a V U d\xi \\
& + \int_{\ell} m_a V \frac{\partial w_z}{\partial \xi} \sin\theta\, \xi d\xi \\
& - \int_{\ell} m_a U \frac{\partial w_z}{\partial \xi} \cos\theta\, \xi d\xi \\
& + \int_{\ell} a \rho g A \cos\theta\, \xi d\xi
\end{aligned}$$

The only additional moments are the buoyancy moments. All other moments are considered to be zero for the specific problem considered in this report.

APPENDIX B
COMPUTER PROGRAM DESCRIPTIONS

OVERVIEW

The equations of motions developed in the previous sections of this report have been solved by means of digital computer programs. Two major programs have been developed: the first (MAIN) solves the equations of motion using the Runge-Kutta-Merson integration algorithm and generates time histories that are stored on the system disk. The second (PLTHSP) generates California Computer Products Company (CALCOMP) pen plots from the disk files. All programs were designed to operate on the Control Data Corporation computer system, located at the David W. Taylor Naval Ship Research and Development Center in Carderock, Md.

Descriptions of input data required to execute the programs, job control cards, and programs follow. Sufficient detail is presented for this appendix to serve as a manual for use and maintenance.

JOB CONTROL CARDS FOR PROGRAM MAIN

Job control cards for program MAIN which computes time histories of the motion variables, are described as follows. If CALCOMP plots are not desired, TAPES need not be cataloged.

Job Control Language Card:	Comment
Job Card	Standard facility card
Charge Card	Standard facility card
REQUEST,TAPE9,*PF.	Reserves space for CALCOMP plot data
REQUEST,TAPE2,*PF.	Print output file 1 request
REQUEST,TAPE4,*PF.	Print output file 2 request
ATTACH,BINAR,SEFZARNICKNEWB, ID=XXXX.	Attaches binary run file
ATTACH,NSRDC.	Attaches library routines
LDSET(LIB=NSRDC).	Loads library routines
BINAR.	Loads and executes run file
REWIND,TAPE2. REWIND,TAPE4.	Rewinds time-history files for printing
COPY(TAPE2,OUTPUT)	Prints time-history file
COPY(TAPE4,OUTPUT)	Prints time-history file

Job Control Language Card:	Comment
CATALOG,TAPE9, SEFZARNICKDATA.., ID=XXXX.	Catalogues file for plot. (SEFZARNICKDATA CAN BE ANY NAME)
7/8/9 END OF RECORD	
DATA CARDS (1-5)	
6/7/8/9 END OF FILE	

INPUT DATA CARDS FOR PROGRAM MAIN

Input data used by program MAIN are read from data cards in NAMELIST and in standard format. A description of the FORTRAN symbols appearing in NAMELIST follows. For simplicity in the text that follows, it is assumed that NAMELIST input occupies only one card. More cards can be used if necessary.

Card 1(NAMELIST FORMAT, /   /)

A	The absolute error for KUTMER (six values)
NPRINT	If=1, print normal output
	If=2, matrix, inverse matrix, F-column matrix, and KUTMER results
	If=3, integral results
	If=4, calculated values constant for given input values
NPLOT	If=0, no plot
	If=1, printer plot of results
END	Number of runs to be made
W	Weight of craft in pounds
BL	Boat length in feet
TZ	Thrust component in z direction
TX	Thrust component in x direction
XECG	Distance from center of gravity to center of pressure for drag force in feet
XP	Moment arm of propeller thrust
XD	Distance from center of gravity to center
DRAG	Friction for drag force
RO	Wave height
LAMBDA	Wavelength
RG	Radius of gyration in feet
T	Propeller thrust in pounds
GAMMA	Propeller thrust angle in degrees

Card 1 (continued)

ECG	Longitudinal center of gravity
NCG	Vertical center of gravity, nondimensionalized by ship length
KAR	Added-mass coefficient
BETA(I)	Dead-rise angle in degrees
EST(I)	Station position in feet
NUM	Number of stations
XA	Initial time
XE	Stop time
HMIN	Minimum step size
HMAX	Maximum step size
EPS	Error criterion

Card 2 (Format 8F10.0)

(X(I),I=1,6)	Initial conditions
X(1)	Velocity
X(2)	Z
X(3)	θ
X(4)	X
X(5)	Z
X(6)	θ degrees

Card 3 (8F10.0)

START	Time to turn on (RMP) function (see page 48)
RISE	Duration of RMP

Card 4 (8F10.0)

TME	Time at which integration interval is to be changed*
HMX	New maximum interval size after TME
HMN	New minimum interval size for KUTMER to subdivide

---

*If this option is not used set TME to stop time on run.

Card 5 (8F10.0)

PERCNT	Percentage of boat length subtracted from longitudinal center of gravity to obtain X - point where acceleration computations are made

JOB CONTROL CARDS FOR PROGRAM PLTHSP

Job control cards for program PLTHSP which generates CALCOMP plots of time histories computed by program MAIN are described in this section.

Job Control Language Card:	Comment
Job Card	Standard facility card
Charge Card	Standard facility card
REQUEST,TAPE7,HI.	Tape for CALCOMP plot data
VSN(TAPE7=CK0323).	Volume serial number of tape for CALCOMP plot
ATTACH,CALC936.	Attaches CALCOMP library routine
ATTACH,BINAR,SFFZARNICKPLOTB, ID=XXXX.	Attaches plot program run file
LDSET(LIB=CALC936)	Loads CALCOMP library routines
BINAR.	Runs plot program
7/8/9 END OF RECORD	
DATA CARDS	
6/7/8/9 END OF FILE	

INPUT DATA CARDS FOR PROGRAM PLTHSP

Two or three data cards are made ready by PLTHSP, depending on the options selected. Standard input format is employed. A description of the necessary data cards follows.

Card 1 (8F10.0 Format)

XAXIS	Length of x axis in inches
YAXISP	Height of pitch component axis in inches
YAXISH	Height of heave component axis in inches
HT	Height of lettering in inches

Card 2 (I10 Format)

IA	If=0, no plots for bow acceleration and center of gravity acceleration If=1, plots previously mentioned information

Card 3 (8F10.0 Format) - Only Necessary If IA = 1.

YAXISB	Height of bow acceleration axis in inches
YAXISC	Height of CG acceleration axis in inches

PROGRAM MAIN

Program MAIN reads all necessary input data from cards, sets up initial values, computes constants, calls KUTMER to determine the state variables at TIME for the period from XA to XE in increments of HMAX. A table state variables is created for every PTIME-th value. The values for λ/H and $\theta_p/2\pi H/\lambda$ are calculated and printed. If the plot option is on, a printer plot will be produced.

Subroutine COMPUT(X)

This routine computes pitch moment NL and lift force FL, excluding added mass terms, using values of integrals computed in subroutine FUNCT. The argument X contains the state vector.

Subroutine DAUX

This subroutine is called from KUTMER or EULER. It determines the values of m_a, b, and b1*, based on the following equations

$$h_w(I) = z_{CG} - \xi(I)\sin\theta + \zeta(I)\cos\theta - r(I)$$

where $r(I) = r_0 \cos k\,[x_{CG} + \xi(I)\cos\theta + \zeta(I)\sin\theta + ct]$

Then for

$$h_w(I) > 0\,,$$

$$d(I) = \frac{h_w(I)}{\cos\theta - (I)\sin\theta}$$

where $V(I) = -r_0 k \sin\theta\,[x_{CG} + \xi(I)\cos\theta + (I)\sin\theta + ct]$

If

$$d(I) \geqslant b_m(I)\tan(\beta(I)\ 2/\pi)$$

set

$$m_a(I) = m_{amax}(I)$$

$$b(I) = b_m(I)$$

$$bl(I) = 0$$

$$m_{amax}(I) = k(I)(\rho/2)\pi b_m^2(I)$$

If

$$d(I) < b_m(I) \tan(\beta(I))(2/\pi)$$

set

$$b(I) = d(I) \cot(\beta(I))(\pi/2)$$

$$bl(I) = b(I)$$

$$m_a(I) = k_a(I)(\rho/2)\pi b^2(I)$$

for

$$h_w(I) < 0;$$

$$m_a(I) = 0, \quad b(I) = 0, \quad bl(I) = 0$$

This subroutine then calls FUNCT which in turn calls COMPUT to determine the values of N_L and F_L, the lift force and moment. The values of N_L and F_L are used to compute the following

$$F_1 = T_x + F_L \sin\theta - D\cos\theta$$

$$F_2 = T_z + F_L \cos\theta + D\sin\theta + W$$

$$F_3 = N_L - D_{x_d} + T_{x_p}$$

---

*bl array is set up for integrations for portion of hull for which chine is not immersed.

The mass inertia matrix is

$$A_{11} = M + M_a \sin^2 \theta$$

$$A_{12} = M_a \sin \theta \cos \theta$$

$$A_{13} = -Q_a \sin \theta$$

$$A_{21} = A_{12}$$

$$A_{22} = M + M_a \cos^2 \theta$$

$$A_{23} = -Q_a \cos \theta$$

$$A_{31} = A_{13}$$

$$A_{32} = A_{23}$$

$$A_{33} = I + I_a$$

The matrix is inverted by the system routine MATINS. The inverted matrix is then used to solve the following equations which determine the state vectors.

$$\ddot{x}_{CG} = A_{11}^{-1} F_1 + A_{12}^{-1} F_2 + A_{13}^{-1} F_3$$

$$\ddot{z}_{CG} = A_{21}^{-1} F_1 + A_{22}^{-1} F_2 + A_{23}^{-1} F_3$$

$$\ddot{\theta} = A_{31}^{-1} F_1 + A_{32}^{-1} F_2 + A_{33}^{-1} F_3$$

Subroutine FUNCT (X)

This routine evaluates various integrals appearing in the force and moment mathematical models. The integrals are evaluated, using a trapezoidal integration algorithm. The argument x contains the state vector. A list of integrals that are evaluated is presented.

$$\int_\ell m_a d\xi \qquad \int_\ell m_a \xi d\xi$$

$$\int_\ell m_a \xi^2 d\xi \qquad \int_\ell m_a UV d\xi$$

$$\int_\ell m_a w_z d\xi \qquad \int_\ell m_a w_z \xi d\xi$$

$$\int_\ell m_a \frac{dw_z}{dt} d\xi \qquad \int_\ell m_a \frac{dw_z}{dt} \xi d\xi$$

$$\int_\ell m_a V \frac{\partial w_z}{\partial \xi} d\xi \qquad \int_\ell m_a V \frac{\partial w_z}{\partial \xi} \xi d\xi$$

$$\int_\ell m_a U \frac{\partial w_z}{\partial \xi} d\xi \qquad \int_\ell m_a U \frac{\partial w_z}{\partial \xi} \xi d\xi$$

$$\int_\ell m_a V d\xi \qquad \int_\ell m_a V \xi d\xi$$

$$\int_\ell b V^2 d\xi \qquad \int_\ell b V^2 \xi d\xi$$

$$\int_\ell b \left(h - \frac{b}{2} \tan \beta\right) d\xi \qquad \int_\ell b \left(h - \frac{b}{2} \tan \beta\right) \xi d\xi$$

Subroutine INPUT

This routine reads in NAMELIST/HSP/ which contains the initial data concerning the craft and sea conditions pertinent to all the runs to be made. It is set up so that most of the data are given default values by means of data statements in subroutine INPUT. These data statements can be overridden during execution by reading values in on cards. For further explanation of the specific variables see section on the input data cards.

This routine also "initializes" constant such as π, ρ, and g. It uses the input values to calculate the keel profile and planform arrays, NO and BM, wave constants, system mass and inertia, and maximum mass and depth of chine at each station.

Subroutine KUTMER (NEQS, TIME, HMAX, X, EPSE, A, HMIN, FIRST)

This is a Runge-Kutta-Merson integration routine that is capable of changing the size of the interval over which it integrates to meet specified error criteria. It is therefore an accurate method for a system that may oscillate more rapidly than the initial integration interval. A minimum step size prevents the routine from subdividing the interval indefinitely.

The input arguments are:

NEQS	Number of dependent variables in the x array
TIME	Actual time (independent variable)
HMAX	Increment for which the solution is to be returned
X	Vector of dependent variables
EPSE	Relative error criteria specified for each component of x and used for the components of x less than the absolute value of A
A	Absolute error criteria
HMIN	Minimum step size allowed
FIRST	Set to zero on first call; a value of 1 is assigned by KUTMER on subsequent calls for which the error criteria are satisfied, otherwise a value of 2 is assigned

Subroutine PLOT2 (F, FMIN, FMAX, NVAR, NFUN, N1, N, XO, DELX)

Data stored in the two-dimensional array F are plotted, using the printer by subroutine PLOT2. As many as 26 different functions, having evenly spaced abscissa values, can be plotted. The output is written on Unit 6. A description of variables follows.

F	Array containing data to be plotted; the Jth point of the Ith function is stored in F(I,J)
FMIN	An array of minimum functional values; the minimum of the Ith function is stored in FMIN(I)
FMAX	Same as FMIN only for maximum values
NVAR	An array of titles for each function to be plotted
NFUN	Number of functions to be plotted
N1	First dimension of array F
N	Number of points to be plotted
XO	First abscissa value
DELX	Abscissa increment

Subroutine PLOTER (FX, XA, HMAX, LAMBDA, IB, NWAVE)

The routine initializes various values required to generate printer plots and computes pitch-and-heave ratios. The printer plots that are generated consists of pitch-and-heave time histories. A description of input variables follows.

FX	A two-dimensional array, containing time histories to be plotted
XA	Initial time
HMAX	Time-interval increment; time interval between values in FX is given by HMAX*PTIME
LAMBDA	Wavelength
IB	Number of values to be plotted
NWAVE	Position in FX at which wave is completely turned on

Function RMP (T, START, RISE)

The RMP is a function that calculates a value between 0 and 1 corresponding to time T, based on a straight line from time START with a value of 0 to time START plus RISE with a value of 1. It is used to lower the initial wave amplitude to avoid large transients at start of the computations.

The arguments are:

T	Actual time
START	Time at which to begin the ramp from 0 to 1
RISE	Duration of rise from 0 to 1

The function reaches the value 1 at time START plus RISE, if the rise is 0.0, RMP will return a value of 0.5.

Subroutine TRAP (F, DX, NPTS, ANS)

This routine performs the evaluation of an integral using a trapezoidal approximation. The argument variables are defined as follows:

F	Array of integrand values
DX	Increments at which F is evaluated
NPTS	Number of values in F
ANS	Result, which is equal to

$$DX\left\{\sum_{I=1}^{NPTS} F(i) - 0.5\,[F(1) + F(NPTS)]\right\}$$

PROGRAM PLTHSP

This program uses a data file created by program MAIN to create CALCOMP plots. The data are read from logical Unit 9 and are rewritten on Unit 7 for CALCOMP input. Program PLTHSP sets the tape output unit equal to 7 and calls SUBROUTINE CALPHI to execute the plot procedures.

Subroutine CALPLT

This subroutine manages all the I/O operations and performs the necessary calculations required to generate the plots. After reading the card data (two or three cards) subroutine READT is called to read the data file (Tape 9) created by program MAIN. The CALCOMP initializing routines are called next, after which a call to subroutine ESCALE calculates the necessary scaling factors. Subroutine EXAXIS is called next to determine the placement of the plot tick marks and identifying digits. The CALCOMP plot-generation subroutines are now called and, depending on the option defined by the IA parameter on card 2, plots of pitch and heave at the bow and CG location are generated as functions of time if IA = 1.

Subroutine EAXIS

The subroutine is analogous to the CALCOMP AXIS routine. The only exception is that the tick marks are not necessarily inch, and the height of the characters is defined by the input parameter HT. Function NDIGIT is called to determine the number of digits necessary to print an even increment of the plots functions on the axis.

Subroutine ESCALE, ADJUST, and FUNCTION UNIT

These subroutines find the scale to be used on the plot axis. Function UNIT is called to determine the axis increment size after which subroutine ADJUST is called to extend the minimum (AMIN) and maximum (AMAX) values so that they are even multiples of the axis increments.

FUNCTION NDIGIT

This function finds the number of digits necessary to print even increments of the function on the axis. Both the number of places in the entire number (NDIGIT) and the number of decimal places (ND) are determined, after which the value of each increment on the axis (ANUM) is calculated.

Subroutine READT

This subroutine reads the data file created by program MAIN. Data file records are read until the message end of file is encountered. Each record is read in the same format as it was written in MAIN. The information is printed to allow the user to inspect the created file.

LISTING OF COMPUTER PROGRAM FOR MOTION COMPUTATIONS

```
      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE3=512,
     *   TAPE2=512,TAPE4=512,TAPE9)
C
      REAL IT,K,LAMBDA,M,MA,MMAX,N,NCG,NU,MASS,NL,IA,KAR
      INTEGER END
C
      DIMENSION X(6),FX(2,400)
C
      COMMON /CONST/ NCG,ECG,PI,DPR,RPD,GRAVTY,RHO,K,NUM,MA(120),CD,TA,
     *         B(120),BETA,HW(120),T2,DRAG,W,XD,T,XP,M,IT,
     *         DELTAS,TX,EST(120),C,RO,KAR,MMAX(1 0),TEST(120),
     *                N(120),PHALF
      COMMON /SHIP/ MASS,CINT,QA,CE,CE2,CE3,DMU,EDMU,E2DMU,E3DMU,BF,BMM,
     *              NL,FL,IA,E(120)
      COMMON /IN/ BM(120),BI(120),VELIN
      COMMON/OUT/NPRINT,NPLOT,END
      COMMON/TERMS/T1,T2,T3,T4,T5,T6,T7,T8
      COMMON /SEAWAVE/ START,RISE,RAMP
      COMMON /INTER/ II,KTT(10),DIFF(10)
      COMMON /IN2/ NO(120),XA,XE,HMAX,HMIN,A(6),EPSE(6),LAMBDA
      COMMON /ACCEL / XACCL,BWACL,CGACL,BL
C
      CALL INPUT
C
C        COMPUTE INTEGRATION INTERVAL INFORMATION
C
      NLESS = NUM-1
      I = 1
      II = 1
      DIFFER = EST(I+1)-EST(I)
      KTT(II) = 1
      DIFF(II) = DIFFER
      DO 25 I=2,NLESS
      DIFFER= EST(I+1)-EST(I)
      KTT(II) = KTT(II)+1
      IF(DIFFER.NE.DIFF(II))GO TO 24
      GO TO 25
   24 II = II+1
      KTT(II) = 1
      DIFF(II) = DIFFER
   25 CONTINUE
      KTT(II) = KTT(II)+1
C * * * * * CHECK IF NUMBER OF INTERVALS EXCEEDS DIMENSION
      IF (II.GT.10) WRITE(6,28) (KTT(I),DIFF(I),I=1,II)
      IF(II.GT.10) STOP 4
C * * * * * POINT AT WHICH MULTIPLE RUNS START
    8 CONTINUE
      TIME=XA
      KOUNT=1
      END=END-1
      WRITE(6,39)
   39 FORMAT(1H1)
C * * * * * * * * READ IN INITIAL CONDITIONS
C    X(1) = VELOCITY,  X(2) = Z DOT,  X(3) = THETA DOT
C     X(4) = X,         X(5) = Z,      X(6) = THETA
C      THETA IS READ IN DEGREES THEN CONVERTED TO RADIANS IN PROGRAM
C
      READ(5,10)(X(I),I=1,6)
C
```

```
C              DATA , USED IN RAMP FUNCTION, TO TURN UN WAVE
      READ(5,10)START,RISE
C
   10 FORMAT(8F1^.4)
C * * * * * * * * WRITE OUT THE INPUT VALUES
      WRITE(6,19) START,RISE,KAR
   19 FORMAT("     START = ",F10.4,/,"   RISE = ",F10.4,/,"   KAH = ",F10
     *.4)
C
C         THE IS THE TIME AT WHICH THE INTEGRATION INTERVAL IS
C             TO HE CHANGED
C         HMX IS THE NEW MAXIMUM INTERVAL SIZE AFTER TIME THE
C         HMN IS THE NEW MINIMUM INTERVAL SIZE FOR KUTMER TU SUB-DIVIDE
C             THE MAXIMUM INTERVAL UP TO
C         IF THIS OPTION IS NOT USED SET THE TU THE STOP TIME OF THE RUN
C
      READ(5,10) THE,HMX,HMN
      WRITE(6,11) THE,HMAX,HMX,HMIN,HMN
   11 FORMAT(*  AT TIME *,F7.2,* THE MAXIMUM INTERVAL SIZE FOR INTEGRATI
     *ON WILL BE CHANGED FROM *,F10.4,* TO *,F10.4,/,
     **  AND THE MINIMUM SIZE FUR HALVING CHANGES FROM *,F10.4,
     *  * TU *,F10.4)
C         ADJUST THE TIME FOR CHANGE OF INTEGRATION INTERVAL
C         FOR CHECK AGAINST TIME IN THE INTEGARTION LOOP
            TH = THE-(HMAX/2.)
C         SET SWITCH FOR CALCULATION UF PITCH AND HEAVE RATIOES
C         ON NEXT CALL TO PLOTER
      IPT = 0
      IF(THE.EQ.XE) IPT = 1
C
      READ(5,10) PERCNT
      XACCL = ECG-PERCNT*BL
      WRITE(6,12) PERCNT,XACCL
   12 FORMAT(*  THE X USED FOR THE BOW AND CG ACCELERATION COMPUTATIONS
     .IS EQUAL TO ECG-*,F10.4,7H*BL OR ,F10.4)
C
      WRITE(6,23)
      WRITE(6,47)
   23 FORMAT(1H ,//)
   47 FORMAT(" STATION NO.",3X,"DEAU RISE",8X,"EST",8X,"NO",
     *  10X,"BEAM")
      WRITE(6,55) ((I,BETA,EST(I),NU(I),BM(I)),I=1,NUM)
   55 FORMAT(6X,I2,5X,F10.4,4X,F10.4,4X,F10.4,3X,F10.4)
      WRITE(6,23)
      WRITE (6,56) (X(I),I=1,6)
   56 FORMAT(" X VALUES",4X,6(F10.4,2X))
C * * * * * * * * CHANGE INPUT FROM DEGREES TO RADIANS
      X(3) = X(3)*RPD
      X(6) = X(6)*RPD
C
      WAVE = START+RISE
      NWAVE = 0
C * * * * * * * * WRITE OUT COMPUTED ARRAYS
      WRITE(6,57)H,IT,K,C,PHALF,PI,GRAVTY
      IF(NPRINT.LT.4)  GO TO 62
      WRITE (6,58) (E(I),I=1,NUM)
      WRITE (6,59) (N(I),I=1,NUM)
      WRITE (6,64) (HMAX(I),I=1,NUM)
      WRITE (6,64) (TEST(I),I=1,NUM)
```

```
   62 CONTINUE
      WRITE(6,28)(KTT(I),DIFF(I),I=1,II)
   28 FORMAT(*  KTT,DIFF *,I10,2X,F10.4)
   57 FORMAT(4H M= ,F10.4,4H I= ,F10.4,4H K= ,F10.4,4H C= ,F10.4,11H PI*
     +RHO/2= ,F10.4,5H PI= ,F10.4,10H GRAVITY= ,F10.4)
   58 FORMAT (" E(I)",10F10.4)
   59 FORMAT (" N(I)",10F10.4)
   64 FORMAT (" HMAX(I)",10F10.4)
   66 FORMAT (" TEST(I)",10F10.4)
      IB = 1
      IPRINT = NPRINT
      WRITE(4,91)
C * * * * * * * * WRITE HEADINGS AND CONDITIONS AT TIME = 0.
   91 FORMAT(1H1,2X,"TIME",9X,"XDOT",9X,"ZDOT",9X,"THETA DOT",6X,
     +  1HX,9X,1HZ,9X,5HTHETA,9X,2HNL,9X,2HFL,
     +  4X,8HBOW ACCL,4X,7HCG ACCL,//)
      WRITE(4,92) TIME,(X(I),I=1,6),NL,FL,BWACL,CGACL
      WRITE(9) TIME,(X(I),I=4,6),BWACL,CGACL
      KOUNT = KOUNT+1
      FX(1,IB)=X(5)
      FX(2,IB)=X(6)
      IKUTM=(XE-XA)/HMAX+.05
      IKUTM = (TME-XA)/HMAX + (XE-TME)/HMX + .05
      FIRST=0.0
      NEQS=6
      IKUTS=0
C
C        START OF INTEGRATION LOOP
C
  851 CONTINUE
      NPRINT = IPRINT
C * * * * * * * * CHECK PITCH .GT. .5236 RADIANS
      IF(X(6).GT..5236)GO TO 853
C * * * * * * * *  PERFORM INTEGRATIONS
      IF(TIME.LT.TM.OR.TME.EQ.XE) GO TO 98
        IF(IPT.EQ.1) GO TO 98
          HMIN = HMN
          HMAX = HMX
          FIRST = 0.0
   98 CONTINUE
      CALL KUTMEQ(NEQS,TIME,HMAX,X,EPSE,A,HMIN,FIRST)
      IKUTS=IKUTS+1
      IF(FIRST.EQ.2)GO TO 861
      IF(KOUNT.NE.1.AND.KOUNT.NE.41) GO TO 99
      WRITE(4,91)
      KOUNT=1
C * * * * * * * *  WRITE OUT TIME INTERVAL RESULTS
   99 WRITE(4,92) TIME,(X(I),I=1,6),NL,FL,BWACL,CGACL
      WRITE(6,93)T1,T2,T3,T4,T5,T6,T7,T8,BMM,BF
      WRITE(9) TIME,(X(I),I=4,6),BWACL,CGACL
      IF(TIME.LT.TM.OR.TME.EQ.XE) GO TO 200
      IF(IPT.EQ.1) GO TO 200
      CALL PLUTEQ(FX,XA,HMAX,LAMBDA,IB,NWAVE,IPT)
      IPT = 1
      I9 = 0
      XA = TIME
      FIRST = 0.0
      HMIN = HMN
      HMAX = HMX
```

```
  200 CONTINUE
      IB=IB+1
      FX(1,IB)=X(5)
      FX(2,IB)=X(6)
   93 FORMAT(" ",10E10.4)
   92 FORMAT(1X,11(F10.4,2X))
  100 CONTINUE
      KOUNT=KOUNT+1
      IF(NWAVE.GT.0)GO TO 21
      IF(TIME.GT.WAVE)NWAVE=KOUNT
   21 CONTINUE
      IF(TIME.LE.XE.AND.IKUTS.LT.IKUTM)GU TO 851
      WRITE(2,85?)
  854 CONTINUE
  852 FORMAT( "   END OF KUTMER")
  853 CONTINUE
      CALL PLUTE?(FX,XA,HMAX,LAMBDA,IB,NWAVE,IPT)
C * * * * * * *  CHECK FOR LAST RUN IF NUT CYCLE HACK TO READ
C     NEW DATA FOR NEXT RUN
      IF(END,NE,1)GU TO 8
      GO TO 999
C * * * * KUTMER ERROR MESSAGES
  861 WRITE(6,86?)
  862 FORMAT("    ERRUR CRITERIUN IN KUTMER CAN NOT BE MET")
      WRITE (6,56) (X(I),I=1,6)
      WRITE (6,86) TIME
   86 FORMAT (" TIME =",F10.4)
      IF(END,NE,1)GO TO 8
      GO TO 853
  999 CONTINUE
      END FILE 9
      END
      SUBROUTINE PLUT2(F,FMIN,FMAX,NVAR,NFUN,N1,N,X0,DELX)
C
C PLUT FIRST N POINTS OF UP TO 26 FUNCTIUNS F(X)
C  F(I,J) CUNTAINS THE VALUE FOR THE JTH POINT OF THE ITH FUNCTION
C  FMIN(I) AND FMAX(I) CONTAIN THE MIN AND MAX ORDINATE VALUES FOR
C          THE ITH FUNCTION.
C       NVAR(I)    AN ARRAY OF TITLES FUR THE VARIOUS FUNCTIONS
C                  TO BE PLOTTED AGAINST THE ABSCISSA
C       NFUN       NUMBER OF FUNCTIUNS TU BE PLOTTED -  DIMENSIUN OF
C                  NVAR, FMIN, FMAX
C       N1         USED ONLY IN F(N1,1) AS PASSED UIMENSION
C       N          NUMBER OF POINTS IN A SINGLE PLOT FRAME
C       X0         FIRST ABSCISSA VALUE
C       DELX       ABSCISSA INCREMENT
C
       DIMENSION ?STEP(26),F(N1,N),FMIN(NFUN),FMAX(NFUN),VLAST(26),
     1        VFI?ST(26),HEAD(6),STEP(26)
        INTEGER CH(26),NVAR(  NFUN),DOT,ASTER,PLUS,BLANK
        INTEGER C
        INTEGER A(101)
C
       DATA BLANK,DOT,ASTER,PLUS/1H ,1H.,1H*,1H+/
       DATA CH(1),CH(2),CH(3),CH(4),CH(5),CH(6),CH(7),CH(8),CH(9),CH(10)
     2   / 1HA , 1HB , 1HC , 1HD , 1HE , 1HF , 1?G , 1HH ,1HI ,1HJ /
       DATA CH(11),CH(12),CH(13),CH(14),CH(15),CH(16),CH(17),CH(18)
     2   /  1HK ,  1HL ,  1HM ,  1HN ,  1HO ,  1HP ,  1HQ ,  1HR/
       DATA CH(19),CH(20),CH(21),CH(22),CH(23),CH(24),CH(25),CH(26)
```

```
     2    / 1HS , 1HT , 1HU , 1HV , 1HW , 1HX , 1HY , 1HZ /      PLOT2 29
C                                                                 PLOT2 30
      IF(NFUN.LE.0.OR.N.LE.0) RETURN                              PLOT2 31
C PRINT HEADINGS.                                                 PLOT2 32
      WRITE(6,46)                                                 PLOT2 33
   46  FORMAT (///)                                               PLOT2 34
      DO 40 I=1,NFUN                                              PLOT2 35
 30    TENM=ABS(FMAX(I)-FMIN(I))                                  PLOT2 36
       EXP=1.                                                     PLOT2 37
       IF (TENM.EQ.0.) GO TO 2                                    PLOT2 38
C ARING TENM TO A VALUE BETWEEN 1 AND 10                          PLOT2 39
       IF(TENM.LT.1.) GO TO 1                                     PLOT2 40
 3     IF(TENM.LT.10.) GO TO 2                                    PLOT2 41
       EXP=EXP*10.                                                PLOT2 42
       TENM=TENM*.1                                               PLOT2 43
       GO TO 3                                                    PLOT2 44
 1     EXP=EXP*.1                                                 PLOT2 45
       TENM=TENM*10.                                              PLOT2 46
       IF(TENM.GT.1.) GO TO 2                                     PLOT2 47
       GO TO 1                                                    PLOT2 48
C  SET UP VALUE BETWEEN GRID LINES, RSTEP.                        PLOT2 49
 2     PSTEP=5.                                                   PLOT2 50
       IF(TENM.GE.5.)PSTEP=10.                                    PLOT2 51
       IF(TENM.LT.2.)PSTEP=2.                                     PLOT2 52
 5     RSTEP(I)=PSTEP*EXP*.1                                      PLOT2 53
C  CUMPUTE VALUE OF STARTING LINE, VFIRST.                        PLOT2 54
       FIRST=FMIN(I)/RSTEP(I)                                     PLOT2 55
       IF(FMIN(I).LT.0.)FIRST=FIRST-1.                            PLOT2 56
       FIRST=AINT(FIRST)                                          PLOT2 57
       VFIRST(I)=FIRST*RSTEP(I)                                   PLOT2 58
C  CHECK END LINE VALUE,VLAST.                                    PLOT2 59
       VLAST(I)=VFIRST(I)+10.*RSTEP(I)                            PLOT2 60
       IF(VLAST(I).GT.FMAX(I))GO TO 4                             PLOT2 61
C  IF GRAPH IS TUO SMALL TAKE NEXT LARGER STEP.                   PLOT2 62
       AA=PSTEP                                                   PLOT2 63
       IF(AA.LT.5.)PSTEP=5.                                       PLOT2 64
       IF(AA.EQ.5.)PSTEP=10.                                      PLOT2 65
       IF(AA.LT.10.) GO TO 5                                      PLOT2 66
       PSTEP=2.                                                   PLOT2 67
       EXP=10.*EXP                                                PLOT2 68
       GO TO 5                                                    PLOT2 69
C  CUMPUTE VALUE BETWEEN POINTS,STEP.                             PLOT2 70
 4     STEP(I)=RSTEP(I)*.1                                        PLOT2 71
       RK=0.                                                      PLOT2 72
      DO 6 KK=1,6                                                 PLOT2 73
       HEAD(KK)=VFIRST(I)+2.*RK*RSTEP(I)                          PLOT2 74
 6     RK=RK+1.                                                   PLOT2 75
  40   WRITE (6,45) CH(I), NVAR(I), (HEAD(KK),KK=1,6)             PLOT2 76
   45 FORMAT(1X,A1,3H = ,A10,5X,1PE12.4,5(8X,1PE12.4))            PLOT2 77
      DO 50 J=1,101                                               PLOT2 78
      A(J)=BLANK                                                  PLOT2 79
      IF(MOD(J,10).EQ.1) A(J)=DOT                                 PLOT2 80
   50 CONTINUE                                                    PLOT2 81
      WRITE(6,55) A,A                                             PLOT2 82
   55  FORMAT (25X,101A1/15X,4HTIME,6X,101A1)                     PLOT2 83
C PLUT EACH PUINT                                                 PLOT2 84
      DO 100 J=1,N                                                PLOT2 85
      B=X0+FLUAT(J-1)*DELX                                        PLOT2 86
      DO 70 K=1,101                                               PLOT2 87
```

```
      A(K)=BLANK                                                        PLOT2 88
      IF(MOD(K,10).EQ.1) A(K)=DOT                                       PLOT2 89
      IF(MOD(J,5 ).EQ.1) A(K)=DOT                                       PLOT2 90
   70 CONTINUE                                                          PLOT2 91
      DO 80 I=1,NFUN                                                    PLOT2 92
       LOC=((F(I,J)-VFIRST(I))/STEP(I)+1.5)                             PLOT2 93
       C=A(LOC)                                                         PLOT2 94
       A(LOC)=CH(I)                                                     PLOT2 95
      IF(C.NE.BLANK.AND.C.NE.DOT) A(LOC)=ASTER                          PLOT2 96
   80 CONTINUE                                                          PLOT2 97
       IF(MOD(J,10).EQ.1)GO TO 95                                       PLOT2 98
      WRITE(6,85) A                                                     PLOT2 99
   85  FORMAT (25X,101A1)                                               PLOT2100
       GO TO 100                                                        PLOT2101
   95 WRITE(6,15)B,A                                                    PLOT2102
   15  FORMAT (12X,1PE12.4,1X,101A1)                                    PLOT2103
  100 CONTINUE                                                          PLOT2104
      RETURN                                                            PLOT2105
      END                                                               PLOT2106
      SUBROUTINE KUTMER(ND,T,H,Y0,EPSE,A,HCX,FIRST)                     KUTMER 2
      DIMENSION Y0(6),Y1(6),Y2(6),F0(6),F1(6),F2(6),EPSE(6),A(6)        KUTMER 3
      COMMON/OUT/NPRINT,NPLOT,END                                       KUTMER 4
      COMMON /ACCEL / XACCL,BWACL,CGACL,BL                              KUTMER 5
      DATA NAM1,NAM2 /2HY1,2HY2 /                                       KUTMER 6
C                                                                       KUTMER 7
C        ND = NUMBER OF EQUATIONS, NO. OF COMPONENTS OF Y0              KUTMER 8
C        T  = INDEPENDENT VARIABLE                                      KUTMER 9
C        H  = INCREMENT FOR WHICH SOLUTION IS TO BE RETURNED + OR -     KUTMER10
C        Y0 = THE VECTOR OF DEPENDENT VARIABLES.  ENTER WITH INITIAL    KUTMER11
C     VALUES AT T AND RETURN WITH VALUES AT T+H                         KUTMER12
C      EPSE = RELATIVE ERROR CRITERION FOR COMPONENTS OF Y0 .GT ABS(A)  KUTMER13
C        A  = ABSOLUTE ERROR SRITERION FOR COMPONENTS OF Y0 .LT. ABS(A) KUTMER14
C  NOTE-- EPSE AND A MUST BE SPECIFIED FOR EACH COMPONENT OF THE SYSTEM KUTMER15
C       HCX = THE SMALLEST STEP SIZE USED IN THE INTEGRATION            KUTMER16
C    FIRST SHOULD BE 0 WHEN KUTMER IS ENTERED FOR THE FIRST TIME        KUTMER17
C    AFTER THAT FIRST IS 1 IF KUTMER IS ENTERED WITH THE SAME H OR      KUTMER18
C       IF IT IS ENTERED WITH A CHANGED H                               KUTMER19
C    IF FIRST IS 2 THE ERROR SRITERIA CANNOT BE MEET AND THE STEP SIZE IKUTMER20
C    REDUCED TO H/128.                                                  KUTMER21
C                                                                       KUTMER22
      IF (FIRST) 20,10,20                                               KUTMER23
C - - - - - - - - - - - - FIRST ENTRY                                   KUTMER24
   10 HC = H                                                            KUTMER25
      IPLOC = 1                                                         KUTMER26
      FIRST = 1.                                                        KUTMER27
C - - - - - - - - - - -  OTHER ENTRY                                    KUTMER28
   20 LOC = 0                                                           KUTMER29
      HCX = HC                                                          KUTMER30
      IF (HC.NE.0.) GO TO 30                                            KUTMER31
      WRITE(6,800)                                                      KUTMER32
  800 FORMAT(5X,45HKUTMER ENTERED WITH ZERO INTEGRATION INTERVAL  )     KUTMER33
      FIRST = 2.                                                        KUTMER34
      RETURN                                                            KUTMER35
C - - - - - - - - - - - -  5 CALLS TO DAUX                              KUTMER36
   30 CALL DAUX(T,Y0,F0)                                                KUTMER37
      IF(NPRINT.EQ.5)WRITE(6,400)Y0,T,F0                                KUTMER38
  400 FORMAT(6(2X,F10.4),4HTIME,2X,F10.4)                               KUTMER39
      IF(NPRINT.EQ.5)WRITE(6,400)HC                                     KUTMER40
   39 DO 40 I=1,ND                                                      KUTMER41
```

```
   40 Y1(I) = Y0(I)+(HC/3.)*F0(I)                                        KUTMER42
      IF(NPRINT.EQ.5)WRITE(6,400)Y1,T                                   KUTMER43
C                                                                       KUTMER44
      CALL DAUX(T+HC/3.,Y1,F1)                                          KUTMER45
      IF(NPRINT.EQ.5)WRITE(6,400)F1,T                                   KUTMER46
      DO 50 I=1,ND                                                      KUTMER47
   50 Y1(I) = Y0(I)+(HC/6.)*F0(I)+(HC/6.)*F1(I)                          KUTMER48
      IF(NPRINT.EQ.5)WRITE(6,400)Y1,T                                   KUTMER49
C                                                                       KUTMER50
      CALL DAUX(T+HC/3.,Y1,F1)                                          KUTMER51
      IF(NPRINT.EQ.5)WRITE(6,400)F1,T                                   KUTMER52
      DO 60 I=1,ND                                                      KUTMER53
   60 Y1(I) = Y0(I)+(HC/8.)*F0(I)+.375*HC*F1(I)                          KUTMER54
      IF(NPRINT.EQ.5)WRITE(6,400)Y1,T                                   KUTMER55
C                                                                       KUTMER56
      CALL DAUX(T+HC/2.,Y1,F2)                                          KUTMER57
      IF(NPRINT.EQ.5)WRITE(6,400)F2,T                                   KUTMER58
      DO 70 I=1,ND                                                      KUTMER59
   70 Y1(I) = Y0(I)+(HC/2.)*F0(I)-1.5*HC*F1(I)+2.*HC*F2(I)               KUTMER60
      IF(NPRINT.EQ.5)WRITE(6,400)Y1,T                                   KUTMER61
C                                                                       KUTMER62
      CALL DAUX(T+HC,Y1,F1)                                             KUTMER63
      IF(NPRINT.EQ.5)WRITE(6,400)F1,T                                   KUTMER64
      DO 80 I=1,ND                                                      KUTMER65
   80 Y2(I) = Y0(I)+HC/6.*F0(I)+(2./3.)*HC*F2(I)+(HC/6.)*F1(I)           KUTMER66
      IF(NPRINT.EQ.5)WRITE(6,400)Y2,T                                   KUTMER67
      INC = 0                                                           KUTMER68
C - - - - - - - - - - -  CHECK ERROR CRITERIA                            KUTMER69
      DO 110 I=1,ND                                                     KUTMER70
      ZZZ = ABS(Y1(I))-A(I)                                             KUTMER71
      IF (ZZZ) 85,87,87                                                 KUTMER72
C - - - - - - - - - - -  ABSOLUTE ERROR                                  KUTMER73
   85 ERROR = ABS(.2*(Y1(I)-Y2(I)))                                      KUTMER74
      IF (ERROR-A(I)) 100,100,90                                        KUTMER75
C - - - - - - - - - - -   RELATIVE ERROR                                 KUTMER76
   87 ERROR = ABS(.2-.2*Y2(I)/Y1(I))                                     KUTMER77
      IF(ERROR-EPSE(I)) 100,100,90                                      KUTMER78
C - - - - - - - - - - -  SINCE ERROR .GT. ERROR CRITERIA CHECK IF HC.GT.H/KUTMER79
C - - - - - - - - - - -  IF YES THEN HALVE INTERVAL.  OTHERWISE STOP.    KUTMER80
   90 X = 128.*ABS(HC)-ABS(H)                                            KUTMER81
      IF(X) 91,95,95                                                    KUTMER82
C - - - - - - - - - - -  ERROR TOO LARGE                                 KUTMER83
   91 WRITE(6,92)I,T,ERROR,HC                                            KUTMER84
   92 FORMAT(/18H FOR EQUATION NO. I2,27H, THE RELATIVE ERROR AT T = ,  KUTMER85
     .     E15.8, 4H IS ,E15.8,13H STEP SIZE = ,E15.8)                  KUTMER86
      FIRST = 2.                                                        KUTMER87
      RETURN                                                            KUTMER88
C - - - - - - - - - - -  HALVE INTERVAL                                  KUTMER89
   95 HC = HC/2.                                                         KUTMER90
      IPLOC = 2*IPLOC                                                   KUTMER91
      LOC = 2*LOC                                                       KUTMER92
      HCX = HC                                                          KUTMER93
      WRITE(2,710)T,I,ERROR,HC                                          KUTMER94
  710 FORMAT(/8H TIME = ,F10.3,5X,26HHALVE INTERVAL.  EQUATION  ,I3,    KUTMER95
     .13H HAS ERROR = ,E16.8,6X,17H STEP SIZE NOW = ,E15.8)             KUTMER96
      WRITE(2,720) NAM2,(Y2(J),J=1,ND)                                  KUTMER97
      WRITE(2,720) NAM1,(Y1(J),J=1,ND)                                  KUTMER98
  720 FORMAT( 2X,A2 / 3(10E13.5/))                                       KUTMER99
      GO TO 30                                                          KUTME100
```

```
C - - - - - - - - - - -  TEST IF INTEHVAL LENGTH CAN BE DOUBLED             KUTME101
  100 IF (ERROR*64.-EPSE(I)) 110,110,101                                    KUTME102
  101 INC = 1                                                               KUTME103
  110 CONTINUE                                                              KUTME104
C - - - - - - - - - - -  UPDATE T AND SOLUTION                               KUTME105
  111 T = T+HC                                                              KUTME106
      DO 112 I=1,ND                                                         KUTME107
  112 Y0(I) = Y2(I)                                                         KUTME108
C - - - - - - - - - - -  GET SOLUTION IN NEXT INTERVAL                       KUTME109
      LOC = LUC+1                                                           KUTME110
      IF (LUC-IPLOC) 120,210,210                                            KUTME111
  120 IF(INC)210,130,210                                                    KUTME112
  130 IF (LUC-(LOC/2)*2) 210,140,210                                        KUTME113
  140 IF(IPLOC-1)210,210,200                                                KUTME114
C - - - - - - - - - - -  DOUBLE INTERVAL LENGTH                              KUTME115
  200 HC = 2.*HC                                                            KUTME116
      LOC = LUC /2                                                          KUTME117
      IPLOC = IPLOC/2                                                       KUTME118
  210 IF(IPLOC-LOC) 30,329,30                                               KUTME119
  329 BWACL = F0(2)-XACCL*F0(3)                                             KUTME120
      COACL = F0(2)                                                         KUTME121
      RETURN                                                                KUTME122
      END                                                                   KUTME123
      END                                                                   KUTME124
      SUBROUTINE DAUX(TIME,X,RHS)                                           DAUX   2
C                                                                           DAUX   3
C        TIME      TIME AT WHICH SYSTEM IS TO BE EVALUATED                  DAUX   4
C        X         STATE VECTOR                                             DAUX   5
C        RHS       THE RIGHT HAND SIDE OF THE EQUATION  S = F A             DAUX   6
C                                                                           DAUX   7
      REAL KAH                                                              DAUX   8
      REAL IA,IT,M,K,MA,MASS,NCG,NL,N,MMAX                                  DAUX   9
      INTEGER END,PTIME                                                     DAUX  10
      DIMENSION X(6),RHS(6),F(3,1),A(3,3),INDEX(3,3),                       DAUX  11
     *           R(120),V(120),D(120)                                       DAUX  12
C                                                                           DAUX  13
      COMMON /SHIP/ MASS,CINT,QA,CE,CE2,CE3,DMU,EDMU,E2DMU,E3DMU,BF,BMH,DAUX  14
     *           NL,FL,IA,E(120)                                            DAUX  15
      COMMON /CONST/ NCG,ECG,PI,DPR,RPD,GRAVTY,RHO,K,NUM,MA(120),CD,TA, DAUX  16
     *       B(120),BETA,HW(120),TZ,DHAG,W,XD,T,XP,M,IT,                    DAUX  17
     *       DELTAS,TX,EST(120),C,RO,KAR,MMAX(1.0),TEST(120),               DAUX  18
     *            N(120),PHALF                                              DAUX  19
      COMMON /IN/ BM(120),BI(120),VELIN                                     DAUX  20
      COMMON/OUT/NPRINT,NPLOT,END                                           DAUX  21
      COMMON /SEAWAVE/ START,RISE,RAMP                                      DAUX  22
      COMMON /WAVE/ R,PT(120),ZMA,ZWMA,EMAS,ZZWMA,ZWEMA,Z2WMA,E2MAZ,        DAUX  23
     *           ZWDOT(120)                                                 DAUX  24
C                                                                           DAUX  25
      RAMP = RMP(TIME,START,RISE)                                           DAUX  26
      PIH = PI/2.                                                           DAUX  27
      CT = C*TIME                                                           DAUX  28
      CX6 = COS(X(6))                                                       DAUX  29
      SX6 = SIN(X(6))                                                       DAUX  30
C*******SET VALUES OF MA AND B                                              DAUX  31
      DO 75  I=1,NUM                                                        DAUX  32
      PT(I) = (X(4)+E(I)*CX6+N(I)*SX6+CT)*K                                 DAUX  33
      R(I) = RO*COS(PT(I))*RAMP                                             DAUX  34
C * * * * * * * *  COMPUTE HW SUBMERGENCE OF A POINT AND R  THE WAVE        DAUX  35
C              HW(I) IS IN THE FIXED COORDINATE SYSTEM                      DAUX  36
```

```
      HW(I) = X(5)-E(I)*SX6+N(I)*CX6-R(I)
      IF(HW(I).GT.0)  GO TO 65
C                CRAFT IS NOT SUHMERGED
      MA(I) = 0.
      B1(I)=0.
      B(I) = 0.
      GO TO 75
   65 V(I) = -RU*K*SIN(PT(I))*RAMP
      D(I) = HW(I)/(CX6-V(I)*SX6)
C               D(I) IS IN THE BODY AXIS SYSTEM AND IS THE SUBMERGENCE
      IF(D(I).GE.TEST(I))  GO TU 70
C                CRAFT IS PARTLY SUBMERGEU
      B(I) = D(I)*(1./TA)*PIH
      B1(I) = D(I)*(1./TA)*PIH
      MA(I) = KAR*PHALF*R(I)*B(I)
      GO TO 75
C                CHINE IS IMMERSED
C                     B1 ARRAY IS USED FOR THE INTEGRALS OVER THE PORTION
C                     OF THE HULL FOR WHICH THE CHINE IS NOT IMMERSED
   70 MA(I)=MMAX(I)
      B(I)=BM(I)
      B1(I)=0.
   75 CONTINUE
      IF(NPRINT.LT.4)  GO TO 85
      WRITE(6,74)TIME
   74 FORMAT("  TIME = ",F10.4)
      WRITE(6,76)  (X(I),I=1,6)
      WRITE(6,77) (R(I),I=1,NUM)
      WRITE (6,78) (HW(I),I=1,NUM)
      WRITE (6,79) ( B(I),I=1,NUM)
      WRITE (6,80) (V(I),I=1,NUM)
      WRITE (6,81) (D(I),I=1,NUM)
      WRITE(6,82)(MA(I),I=1,NUM)
   76 FORMAT("  X(I) ",6(2X,E12.6))
   77 FORMAT (" R(I)",10F10.4)
   78 FORMAT (" HW(I)",10F10.4)
   79 FORMAT (" A(I)",10F10.4)
   80 FORMAT (" V(I)",10F10.4)
   81 FORMAT (" D(I)",10F10.4)
   82 FORMAT(" MA(I) ",10F10.4)
   85 CONTINUE
C
C * * * * * * * * CUMPUTES NL AND FL ANU THE ASSOCIATED INTERGALS
      CALL FUNCT(X)
C
      IF(NPRINT.LT.4)GO TO 17
      WRITE(6,15) TX,FL,DRAG,TZ,W,NL,XD,T,XP
   15 FORMAT(" ",10E12.6)
   17 CONTINUE
C * * * * * * * COMPUTE THE F VECTOR
      F(1,1) = TX+FL*SX6-DRAG*CX6
      F(1,1)=0.0
      F(2,1) = TZ+FL*CX6+DRAG*SX6+W
      F(3,1)=NL-DRAG*XD+T*XP
      IF(NPRINT.LT.3)GO TO 18
      WRITE(6,10)(F(I,1),I=1,3)
   18 CONTINUE
C * * * * * * * COMPUTE THE A MATRIX
      A(1,1) = M+MASS*SX6*SX6
```

```
      A(1,2) = MASS*SX6*CX6                                             DAUX  96
      A(1,3) = -QA*SX6                                                  DAUX  97
      A(1,2) = 0.                                                       DAUX  98
      A(1,3) = 0.                                                       DAUX  99
      A(2,1)=A(1,2)                                                     DAUX 100
      A(2,2) = M+MASS*CX6*CX6                                           DAUX 101
      A(2,3) = -QA*CX6                                                  DAUX 102
      A(3,1)=A(1,3)                                                     DAUX 103
      A(3,2)=A(2,3)                                                     DAUX 104
      A(3,3)=IT+IA                                                      DAUX 105
      IF(NPRINT.LT.3)GO TO 25                                           DAUX 106
      WRITE(6,12)(A(I,1),I=1,3)                                         DAUX 107
      WRITE(6,13)(A(I,2),I=1,3)                                         DAUX 108
      WRITE(6,14)(A(I,3),I=1,3)                                         DAUX 109
C * * * * * * * INVERT THE A MATRIX                                     DAUX 110
   25 CALL MATINS(A,3,3,F,1,1,DETERM,ID,INDEX)                          DAUX 111
      IF(ID.EQ.2)WRITE(6,26)                                            DAUX 112
   26 FORMAT("     MATRIX IS SINGULAR          ")                       DAUX 113
C*******A ON RETURN WILL CONTAIN THE INVERSE MATRIX                     DAUX 114
C         ID=2 MATRIX IS SINGULAR                                       DAUX 115
C           =1 INVERSE WAS FOUND                                        DAUX 116
C                                                                       DAUX 117
C * * * * * * * *  COMPUTE THE RIGHT HAND SIDE                          DAUX 118
      RHS(1) = F(1,1)                                                   DAUX 119
      RHS(2) = F(2,1)                                                   DAUX 120
      RHS(3) = F(3,1)                                                   DAUX 121
      RHS(1) = 0.0                                                      DAUX 122
      RHS(4) = X(1)                                                     DAUX 123
      RHS(5) = X(2)                                                     DAUX 124
      RHS(6) = X(3)                                                     DAUX 125
   10 FORMAT("    F(I,1) ",3(2X,E12.4))                                 DAUX 126
   12 FORMAT("    A(I,1) ",3(2X,E12.4))                                 DAUX 127
   13 FORMAT("    A(I,2) ",3(2X,E12.4))                                 DAUX 128
   14 FORMAT("    A(I,3) ",3(2X,E12.4))                                 DAUX 129
   30 IF(NPRINT.LT.2) GO TO 40                                          DAUX 130
      WRITE(6,12)  (A(I,1),I=1,3)                                       DAUX 131
      WRITE(6,13)  (A(I,2),I=1,3)                                       DAUX 132
      WRITE(6,14)  (A(I,3),I=1,3)                                       DAUX 133
      WRITE(6,35)  (RHS(I),I=1,6)                                       DAUX 134
   35 FORMAT("    RHS(I) ",6(2X,E12.6))                                 DAUX 135
   40 CONTINUE                                                          DAUX 136
      RETURN                                                            DAUX 137
      END                                                               DAUX 138
      SUBROUTINE FUNCT(X)                                               FUNCT  2
      REAL KAR                                                          FUNCT  3
      REAL IA,IAA,IPART,K,KPI,MA,MASS,NL,NCG,IT,M,MMAX,N                FUNCT  4
      INTEGER END                                                       FUNCT  5
      DIMENSION IPART(120),C1(120),C2(120),                             FUNCT  6
     *           D1(120),D2(120),D3(120),D4(120),D5(120),D6(120),       FUNCT  7
     *           QPART(120),Z1(120),Z2(120),Z3(120),Z4(120),Z5(120),    FUNCT  8
     *              Z6(120),Z7(120)                                     FUNCT  9
     *           ,X(6),VMAA(120)                                        FUNCT 10
C                                                                       FUNCT 11
      COMMON /SHIP/ MASS,CINT,QA,CE,CE2,CE3,DMU,EDMU,E2DMU,E3DMU,BF,BMM,FUNCT 12
     *              NL,FL,IA,E(120)                                     FUNCT 13
      COMMON /CONST/ NCG,ECG,PI,DPR,R2D,GRAVTY,RHO,K,NUM,MA(120),CD,TA, FUNCT 14
     *          B(120),BETA,HW(120),T2,DRAG,W,XD,T,XP,M,IT,             FUNCT 15
     *          DELTAS,TX,EST(120),C,RO,KAR,MMAX(1 0),TEST(120),        FUNCT 16
     *              N(120),PHALF                                        FUNCT 17
```

```
      COMMON /IN/ BM(120),B1(120),VELIN
      COMMON/OUT/NPRINT,NPLOT,END
      COMMON /WAVE/ R(120),PT(120),ZMA,ZWMA,EMAS,ZZWMA,ZWEMA,Z2WMA,E2MAZ
     *                 ,ZWDOT(120)
      COMMON /INTER/ II,KTT(10),DIFF(10)
      COMMON /SEAWAVE/ START,RISE,RAMP
      COMMON /TEST/ VMA
C * * * * * * * * * * * INITIALIZE INTEGRAL SUMS
      MASS = 0.0
      QA = 0.0
      IA = 0.0
      CE = 0.0
      CE2 = 0.0
      DMU = 0.0
      EDMU=0.0
      E2DMU = 0.0
      E3DMU = 0.0
      WF = 0.0
      BMM = 0.0
      ZMA = 0.0
      ZWMA = 0.0
      EMAS = 0.0
      ZZWMA = 0.0
      ZWEMA = 0.0
      Z2WMA = 0.0
      E2MAZ = 0.0
      VPART = X(1)*SIN(X(6))+X(2)*COS(X(6))
      SX6 = SIN(X(6))
      CX6 = COS(X(6))
      W0 = K*C
C * * * * * * * * * SET UP THE FUNCTIONS FOR THE INTEGRALS   (PAGE 4 OF NO
      DO 90 I=1,NUM
      IPART(I)=E(I)*E(I)*MA(I)
      QPART(I)=E(I)*MA(I)
      ZWDOT(I) = -RU*W0*SIN(PT(I))*RAMP
      U = X(1)*CX6-X(2)*SX6+ZWDOT(I)*SX6
      VEL = VPART-X(3)*E(I)-ZWDOT(I)*CX6
      Z1(I) = MA(I)*ZWDOT(I)
      Z2(I) = -MA(I)*COS(PT(I))*RAMP
      Z3(I) = E(I)*Z2(I)
      Z4(I) = E(I)*Z1(I)
      Z5(I) = U*Z2(I)
      Z6(I) = E(I)*Z5(I)
      Z7(I) = MA(I)*VEL*U
      IF (VEL.LE.0.) GO TO 60
      IF (B1(I).LE.0.0) GO TO 50
      DRDT = ZWDOT(I)*(X(1)+C*X(3)*(N(I)*CX6-E(I)*SX6))/C
      D1(I) =     VEL*B1(I)*(X(2)-X(3)*(CX6*E(I)+SX6*N(I)) -DRDT)
      GO TO 51
   50 D1(I) = 0.
   51 CONTINUE
      D2(I) = E(I)*D1(I)
      C1(I) = VEL*VEL*B(I)
      C2(I) = E(I)*C1(I)
      GO TO 61
   60 D1(I) = 0.
      D2(I) = 0.
      C1(I) = 0.
      C2(I) = 0.
```

```
   61 CONTINUE
      D3(I) = Z2(I)*VEL
      D4(I) = E(I)*D3(I)
      PIH = PI/2.
      D5(I) = B(I)*(HW(I)-B(I)*TA/2.)
   66 D6(I) = D5(I)*E(I)*.5
   90 CONTINUE
      RHOG=RHO*GRAVTY
C * * * * * * * SET UP THE FUNCTIONS FOR THE INTEGRALS (PAGE 5 OF NOTES)
      PIH = PI/2.
      KPI = KAR*PI
C    EVALUATE INTEGRALS USING TRAP METHOD
      I = 1
      INDEX = 1
   91 CALL TRAP(MA(INDEX),DIFF(I),KTT(I),TMASS)
      CALL TRAP(QPART(INDEX),DIFF(I),KTT(I),QA1)
      CALL TRAP(C1(INDEX),DIFF(I),KTT(I),CEA)
      CALL TRAP(C2(INDEX),DIFF(I),KTT(I),CE2A)
      CALL TRAP(IPART(INDEX),DIFF(I),KTT(I),IAA)
      CALL TRAP(D1(INDEX),DIFF(I),KTT(I),DMUA)
      CALL TRAP(D2(INDEX),DIFF(I),KTT(I),EDMUA)
      CALL TRAP(D3(INDEX),DIFF(I),KTT(I),E2DMUA)
      CALL TRAP(D4(INDEX),DIFF(I),KTT(I),E3DMUA)
      CALL TRAP(D5(INDEX),DIFF(I),KTT(I),BFA)
      CALL TRAP(D6(INDEX),DIFF(I),KTT(I),BMMA)
      CALL TRAP(Z1(INDEX),DIFF(I),KTT(I),ZMAA)
      CALL TRAP(Z2(INDEX),DIFF(I),KTT(I),ZWMAA)
      CALL TRAP(Z3(INDEX),DIFF(I),KTT(I),EMASA)
      CALL TRAP(Z4(INDEX),DIFF(I),KTT(I),ZZWMAA)
      CALL TRAP(Z5(INDEX),DIFF(I),KTT(I),ZWEMAA)
      CALL TRAP(Z6(INDEX),DIFF(I),KTT(I),Z2WMAA)
      CALL TRAP(Z7(INDEX),DIFF(I),KTT(I),E2MAZA)
C
   93 CONTINUE
      MASS = MASS + TMASS
      QA = QA + QA1
      IA = IA + IAA
      CE = CE + CEA
      CE2 = CE2 + CE2A
      DMU = DMU + DMUA
      EDMU = EDMU + EDMUA
      E2DMU = E2DMU + E2DMUA
      E3DMU = E3DMU + E3DMUA
      BF = BF + RHOG*BFA
      BMM = BMM + RHOG*BMMA
      ZMA = ZMA+ZMAA
      ZWMA = ZWMA+ZWMAA
      EMAS = EMAS+EMASA
      ZZWMA = ZZWMA+ZZWMAA
      ZWEMA = ZWEMA+ZWEMAA
      Z2WMA = Z2WMA+Z2WMAA
      E2MAZ = E2MAZ+E2MAZA
   94 CONTINUE
      IF ( I.GE.II)GO TO 92
      INDEX = INDEX+KTT(I)-1
      I = I+1
      GO TO 91
   92 CONTINUE
C
```

```
C * * * * * * * * CALL COMPUT TO FIND THE VALUE OF VL AND FL USING        FUNCT136
C    THE VALUES OF THE ABOVE INTEGRALS                                    FUNCT137
      CALL COMPUT(X)                                                      FUNCT138
C                                                                         FUNCT139
      IF(NPRINT.LT.3) GO TO 111                                           FUNCT140
      IF(NPRINT.EQ.3) GO TO 108                                           FUNCT141
      IF(NPRINT.EQ.4)GO TO 108                                            FUNCT142
      WRITE(6,97)  (IPART(I),I=1,NUM)                                     FUNCT143
      WRITE(6,98)  (QPART(I),I=1,NUM)                                     FUNCT144
      WRITE(6,99)  (C1(I),I=1,NUM)                                        FUNCT145
      WRITE(6,100)  (C2(I),I=1,NUM)                                       FUNCT146
      WRITE(6,101)  (C3(I),I=1,NUM)                                       FUNCT147
      WRITE(6,102)  (D1(I),I=1,NUM)                                       FUNCT148
      WRITE(6,103)  (D2(I),I=1,NUM)                                       FUNCT149
      WRITE(6,104)  (D3(I),I=1,NUM)                                       FUNCT150
      WRITE(6,105)  (D4(I),I=1,NUM)                                       FUNCT151
      WRITE(6,106)  (D5(I),I=1,NUM)                                       FUNCT152
      WRITE(6,112)  (D6(I),I=1,NUM)                                       FUNCT153
      WRITE(6,113)(Z1(I),I=1,NUM)                                         FUNCT154
      WRITE(6,114)(Z2(I),I=1,NUM)                                         FUNCT155
      WRITE(6,115)(Z3(I),I=1,NUM)                                         FUNCT156
      WRITE(6,116)(Z4(I),I=1,NUM)                                         FUNCT157
      WRITE(6,118)(Z5(I),I=1,NUM)                                         FUNCT158
      WRITE(6,119)(Z6(I),I=1,NUM)                                         FUNCT159
      WRITE(6,120)(Z7(I),I=1,NUM)                                         FUNCT160
      WRITE(6,107)KPI,RHOG,PIH                                            FUNCT161
  108 WRITE(6,109) MASS,CINT,QA,CE,CE2,CE3                                FUNCT162
      WRITE(6,121)IA                                                      FUNCT163
  121 FORMAT(*  IA  *,E10.4)                                              FUNCT164
      WRITE(6,110)DMU,EDMU,E2DMU,E3DMU,BF,BMM                             FUNCT165
      WRITE(6,117)ZMA,ZWMA,EMAS,ZZWMA,ZWEMA,ZZWMA,E2MAZ                   FUNCT166
C  * * * * * * * * * * * FORMATS  * * * * * * * * * * *                   FUNCT167
   96 FORMAT("  CPART(I)",10(2X,E10.4))                                   FUNCT168
   97 FORMAT("  IPART(I)",10(2X,E10.4))                                   FUNCT169
   98 FORMAT("  QPART(I)",10(2X,E10.4))                                   FUNCT170
   99 FORMAT("  C1      ",10(2X,E10.4))                                   FUNCT171
  100 FORMAT("  C2      ",10(2X,E10.4))                                   FUNCT172
  101 FORMAT("  C3      ",10(2X,E10.4))                                   FUNCT173
  102 FORMAT("  D1      ",10(2X.E10.4))                                   FUNCT174
  103 FORMAT("  D2      ",10(2X,E10.4))                                   FUNCT175
  104 FORMAT("  D3      ",10(2X,E10.4))                                   FUNCT176
  105 FORMAT("  D4      ",10(2X,E10.4))                                   FUNCT177
  106 FORMAT("  D5      ",10(2X,E10.4))                                   FUNCT178
  112 FORMAT("  D6      ",10(2X,E10.4))                                   FUNCT179
  107 FORMAT(" KPHI ",E10.4,"RHOG ",E10.4," PHIH ",E10.4)                 FUNCT180
  109 FORMAT(" MASS ",E10.4," CINT ",E10.4," QA ",E10.4," CE ",E10.4,     FUNCT181
     *"CE2 ",E10.4," CE3 ",E10.4)                                         FUNCT182
  110 FORMAT(" DMU ",E10.4," EDMU ",E10.4," E2DMU ",E10.4," E3DMU ",      FUNCT183
     *E10.4," BF ",E10.4," BMM ",E10.4)                                   FUNCT184
  113 FORMAT(4H Z1 ,10(2X,E10.4))                                         FUNCT185
  114 FORMAT(4H Z2 ,10(2X,E10.4))                                         FUNCT186
  115 FORMAT(4H Z3 ,10(2X,E10.4))                                         FUNCT187
  116 FORMAT(4H Z4 ,10(2X,E10.4))                                         FUNCT188
  118 FORMAT(4H Z5 ,10(2X,E10.4))                                         FUNCT189
  119 FORMAT(4H Z6 ,10(2X,E10.4))                                         FUNCT190
  120 FORMAT(4H Z7 ,10(2X,E10.4))                                         FUNCT191
  117 FORMAT(5H ZMA ,E10.4,6H ZWMA ,E10.4,6H EMAS ,E10.4,                 FUNCT192
     *       7H ZZWMA ,E10.4,7H ZWEMA ,E10.4,7H ZZWMA ,E10.4,             FUNCT193
     *         7H E2MAZ ,E10.4)                                           FUNCT194
```

```
  111 CONTINUE                                                          FUNCT195
      RETURN                                                            FUNCT196
      END                                                               FUNCT197
      SUBROUTINE COMPUT(X)                                              COMPUT 2
      DIMENSION X(6)                                                    COMPUT 3
      REAL KAR,KPI                                                      COMPUT 4
      REAL NL,MASS,NCG,M,IT,IA,K,MA,MMAX,N                              COMPUT 5
      INTEGER END                                                       COMPUT 6
C                                                                       COMPUT 7
      COMMON /SHIP/ MASS,CINT,QA,CE,CE2,CE3,DMU,EDMU,E2DMU,E3DMU,BF,BMM,COMPUT 8
     *             NL,FL,IA,E(120)                                      COMPUT 9
      COMMON /CONST/ NCG,ECG,PI,DPR,RPD,GRAVTY,RHO,K,NUM,MA(120),CD,TA, COMPUT10
     *         B(120),BETA,HW(120),TZ,DRAG,W,XD,T,XP,M,IT,              COMPUT11
     *         DELTAS,TX,EST(120),C,RO,KAR,MMAX(1 0),TEST(120),         COMPUT12
     *              N(120),PHALF                                        COMPUT13
      COMMON/OUT/NPRINT,NPLOT,END                                       COMPUT14
      COMMON /TERMS/ T1,T2,T3,T4,T5,T6,T7,T8                            COMPUT15
      COMMON /WAVE/ R(120),PT(120),ZMA,ZWMA,EMAS,ZZWMA,ZWEMA,Z2WMA,     COMPUT16
     *             E2MAZ,ZWDOT(120)                                     COMPUT17
      COMMON /TEST/ VMA                                                 COMPUT18
C                                                                       COMPUT19
C                                                                       COMPUT20
      CX6 = COS(X(6))                                                   COMPUT21
      SX6 = SIN(X(6))                                                   COMPUT22
      WO = K*C                                                          COMPUT23
      PIH = PI/2.0                                                      COMPUT24
      KPI = KAR*PI                                                      COMPUT25
      CONS1 = RO*WO*WO*CX6                                              COMPUT26
      CONS2 = (KPI*RHO*PIH/TA)/CX6                                      COMPUT27
      CONS3 = RO*WO*K*CX6*SX6                                           COMPUT28
      CONS4 = RO*WO*K*CX6*CX6                                           COMPUT29
      TERM1 = X(1)*CX6                                                  COMPUT30
      TERM2 = X(2)*SX6                                                  COMPUT31
      UVNUM = (X(1)*CX6-(X(2)-ZWDOT(NUM))*SX6)*                         COMPUT32
     *        (X(1)*SX6-X(3)*E(NUM)+(X(2)-ZWDOT(NUM))*CX6)              COMPUT33
C                                                                       COMPUT34
      ZMA = ZMA*X(3)*SX6                                                COMPUT35
      ZZWMA = ZZWMA*X(3)*SX6                                            COMPUT36
      ZWMA = ZWMA*CONS1                                                 COMPUT37
      EMAS = EMAS*CONS1                                                 COMPUT38
      DMU = DMU*CONS2                                                   COMPUT39
      EDMU = EDMU*CONS2                                                 COMPUT40
      CE = CE*CD*RHO                                                    COMPUT41
      CE2 = CE2*CD*RHO                                                  COMPUT42
      E2DMU = E2DMU*CONS3                                               COMPUT43
      E3DMU = E3DMU*CONS3                                               COMPUT44
      ZWEMA = ZWEMA*CONS4                                               COMPUT45
      Z2WMA = Z2WMA*CONS4                                               COMPUT46
C                                                                       COMPUT47
   20 T1 = QA*X(3)*(TERM1-TERM2)                                        COMPUT48
      T1 = T1 + ZZWMA - EMAS                                            COMPUT49
      T2 = EDMU                                                         COMPUT50
      T3 = CE2                                                          COMPUT51
      T4 = MA(NUM)*E(NUM)*UVNUM + E2MAZ + E3DMU - ZZWMA + BMM           COMPUT52
      NL = T1 + T2 + T3 + T4 + BMM                                      COMPUT53
      T5 = MASS*X(3)*(TERM2-TERM1)                                      COMPUT54
      T5 = T5 + ZWMA - ZMA                                              COMPUT55
      T6 = -DMU                                                         COMPUT56
      T7 = -CE                                                          COMPUT57
```

```
      T8 = -MA(NUM)*UVNUM - E2DMU + ZWEMA
      BF = BF/CX6
C
      FL=T5+T6+T7+T8-BF
C
      IF(NPRINT.LT.3)GO TO 30
   25 CONTINUE
      WRITE(6,10)NL,FL
   10 FORMAT(" NL = ",E12.6," FL = ",E12.6)
   30 RETURN
      END
      SUBROUTINE INPUT
C* * * * * * * DEFINITION OF INPUT VARIABLES
C        XA = INITIAL TIME
C        XE = FINAL TIME
C     HMIN = MINIMUM STEP SIZE
C     HMAX = MAXIMUM STEP SIZE
C     EPSE = RELATIVE ERROR CRITERION USED FOR VALUES OF Y GT A
C    EPS = ERROR CRITERION IN KUTMER
C      A = ABSOLUTE ERROR CRITERIA USED IN KUTMER
C       NPRINT = 1  FINAL PRINTOUT
C              = 2 MATRIX INVERSE MATRIX,F COLUMN MATRIX,AND KUTMER
C                    RESULTS
C              = 3 INTEGRAL VALUES
C              = 4 CALCULATED VALUES-CONSTANT FOR GIVEN INPUT VALUES
C      NPLOT = 0   NO PLOT
C            = 1  PRINTER PLOT
C        END = NUMBER OF RUNS
C
C        M = MASS OF CRAFT
C        W = WEIGHT OF CRAFT
C       TZ = THRUST COMPONENT IN Z DIRECTION
C       TX = THRUST COMPONENT IN X DIRECTION
C    XECG = DISTANCE FROM CG TO CENTER OF PRESSURE FOR NORMAL FORCE
C       XP = MOMENT ARM OF PROPELLER THRUST
C       XD = DISTANCE FROM CC  TO CENTER OF PRESSURE FOR DRAG FORCE
C     KA(I)= ADDED MASS COEFFICIENT
C        AN ARRAY GIVEN THE VALUE KAR WHICH IS READ IN
C     BM(I) = BEAM AT FREE SURFACE OR AT CHINE
C     DRAG = FRICTION DRAG
C        K = WAVE NUMBER
C       RO = WAVE HEIGHT
C       NU = WAVE SLOPE
C      NUM = NUMBER OF STATIONS
C        BL = BOAT LENGTH
C     LAMBDA = WAVE LENGTH
C        RG = RADIUS OF GENERATION IN FEET
C        T = PROPELLED THRUST IN LBS
C     GAMMA = PROPELLER THRUST ANGLE IN DEGREES
C     DELTAS=STATION SPACING IN FEET
C       ECG = LONGITUDINAL CENTER OF GRAVITY
C       NCG = VERTICAL CG
C     BETA(I) = DEAD RISE
C     NO(I) = HEIGHT OF MEAN BUTTOCK
C       RHO = DENSITY OF WATER
C     GRAVTY = GRAVITY FT/SEC**2
C      DPR = DEGREES PER RADIAN
C       RPD = RADIANS PER DEGREE
C        PI = 3.14159 . . . . . .
```

```
C     EST(I) = STATION POSITION
C     START = START TIME OF THE RAMP FUNCTION FOR SEA WAVE
C       RISE = DURATION OF THE RISE FROM ZERO TO ONE OF THE RAMP
C
C * * * * * * * * IC OPTIONS
C
C          IC(1)   =1 USE WAVE Z DISTANCE IN COMPUTING LIFT COMPONENT
C                    OF NL AND FL
C
C
      REAL IT,K,LAMBDA,M,MA,HMAX,NU,N,NCG,NO,MASS,NL,IA,KAR
      INTEGER END
C
C
      COMMON /CONST/ NCG,ECG,PI,DPR,RPD,GRAVTY,RHO,K,NUM,MA(120),CD,TA,
     *         B(120),BETA,HW(120),TZ,DRAG,W,XD,T,XP,M,IT,
     *         DELTAS,TX,EST(120),C,RO,KAR,HMAX(120),TEST(120),
     *              N(120),PHALF
      COMMON /SHIP/ MASS,CINT,QA,CE,CE2,CE3,DMU,EDMU,E2DMU,E3DMU,BF,BMM,
     *              NL,FL,IA,E(120)
      COMMON /IN/ BM(120),BI(120),VELIN
      COMMON /IN2/ NO(120),XA,XE,HMAX,HMIN,A(6),EPSE(6),LAMBDA
      COMMON/OUT/NPRINT,NPLOT,END
      COMMON /ACCEL/ XACCL,BWACL,CGACL,UL
C
      NAMELIST/HSP/A,NPRINT,NPLUT,END,W,HL,TZ,TX,XECG,XP,XD,
     *                DRAG,RG,T,GAMMA,ECG,NCG,KAR,RO,LAMBDA,NUM,BETA,EST
     *                ,XA,XE,HMIN,HMAX,EPS,VELIN
C
      DATA A /.01,.0001,.00001,.1,.0001,.00001/
      DATA NPRINT,NPLUT,END/1,1,1/
      DATA W,BL,TZ,TX,XECG,XP,XD,DRAG,RO,LAMBDA,RG,T,GAMMA,
     *     ECG,NCG,KAR /16.,3.75,6*0.0,.0416,22.5,.9562,2*0.0,
     *                 2.325,0.0,1.0/
      DATA NUM,BETA,EST /77,20.0,
     *   0.0000,.03125,.06250,.09375,.12500,.15625,.18750,.21875,
     *   .25000,.28125,.31250,.34375,.37500,.40625,.43750,.46875,
     *   .50000,.53125,.56250,.59375,.62500,.65625,.6875,.71875,
     *   .75000,.78125,.81250,.84375,.87500,.90625,.93750,.96875,1.000,
     *   1.06250,1.12500,1.18750,1.25000,1.3125,1.37500,1.4375,
     *   1.500,1.5625,1.625,1.6875,1.75,1.8125,1.875,1.9375,2.0,
     *   2.0625,2.125,2.1875,2.25,2.3125,2.375,2.4375,2.5,2.5625,2.625,
     *   2.6875,2.75,2.8125,2.8750,2.9375,3.0,3.0625,3.125,3.1875,
     *   3.2500,3.3125 ,3.375,3.4375,3.5,3.5625,3.625,3.6875,3.75 /
      DATA XA,XE,HMIN,HMAX,EPS /0.0,20.0,.025,.1,.15/
      DATA VELIN /19.62/
C
C * * * * * * * * READ IN AND WRITE OUT KUTMER PARAMETERS AND PROGRAM
C        OPTIONS
      READ(5,HSP)
      WRITE(6,HSP)
      DO 10 I=1,6
   10 EPSE(I) = EPS
C
C * * * * * * * * SET UP CONSTANTS
      PI = 3.141592653589
      GRAVTY=32.18
      DPR=57.29577951308
      RPD=.017453292519
```

```
      IF(EST(NUM).LT.3.75) STOP 3                                       INPUT109
C                                                                       INPUT110
C        COMPUTE NO AND BM ARRAYS                                       INPUT111
C                                                                       INPUT112
      DO 32 I=1,NUM                                                     INPUT113
      IF(EST(I).GE.0.75) GO TO 30                                       INPUT114
      NO(I)=-0.46875*(1.0-SQRT(EST(I)/0.375-(EST(I)/0.75)**2.0))        INPUT115
      BM(I)=.375*SQRT(1.0-(EST(I)/.75-1.)**2.0)                         INPUT116
      GO TO 32                                                          INPUT117
   30 NO(I)=0.0                                                         INPUT118
      BM(I) = 0.375                                                     INPUT119
   32 CONTINUE                                                          INPUT120
C******COMPUTE CONSTANTS AND INITIALIZE ARRAYS                          INPUT121
      M=W/GRAVTY                                                        INPUT122
      RHO=1.99                                                          INPUT123
      IT=M*RG*RG                                                        INPUT124
      K = 2.*PI/LAMBDA                                                  INPUT125
      C=SQRT(GRAVTY/K)                                                  INPUT126
      NU=RO*K                                                           INPUT127
      PHALF = (PI/2.)*RHO                                               INPUT128
C                                                                       INPUT129
      BETA = BETA*RPD                                                   INPUT130
      CD = COS(BETA)                                                    INPUT131
      TA = TAN(BETA)                                                    INPUT132
      DO 60  I=1,NUM                                                    INPUT133
      E(I) = ECG-EST(I)                                                 INPUT134
      N(I) = NCG+NO(I)                                                  INPUT135
      MMAX(I) = KAR*PHALF*BM(I)*BM(I)                                   INPUT136
      TEST(I) = (2.*BM(I)*TA)/PI                                        INPUT137
   60 CONTINUE                                                          INPUT138
      END=END+1                                                         INPUT139
      RETURN                                                            INPUT140
      END                                                               INPUT141
      SUBROUTINE PLOTER(FX,XA,HMAX,LAMBDA,IB,NWAVE,IPT)                 PLOTER 2
C                                                                       PLOTER 3
C   INPUT:                                                              PLOTER 4
C        FX         A TWO DIMENSIONAL ARRAY CONTAINING PITCH AND        PLOTER 5
C                   HEAVE VALUES AT EACH TIME STEP                      PLOTER 6
C        XA         INITIAL TIME                                        PLOTER 7
C        HMAX       TIME INTERVAL, PTIME*HMAX = INTERVAL BETWEEN        PLOTER 8
C                   FX VALUES                                           PLOTER 9
C        LAMBDA     WAVELENGTH  USED IN CALCULATING PITCH AND           PLOTER10
C                   HEAVE RATIOES                                       PLOTER11
C        IB         NUMBER OF FX VALUES                                 PLOTER12
C        NWAVE      START OF VALUES AFTER WAVE IS COMPLETELY ON         PLOTER13
C                                                                       PLOTER14
C                                                                       PLOTER15
      REAL IT,K,LAMBDA,M,MA,MMAX,N,NCG                                  PLOTER16
      INTEGER END                                                       PLOTER17
C                                                                       PLOTER18
      DIMENSION FX(2,400),FMIN(2),FMAX(2),NVAR(2)                       PLOTER19
C                                                                       PLOTER20
      COMMON /CONST/ NCG,ECG,PI,DPR,RPD,GRAVTY,RHO,K,NUM,MA(120),CD,TA, PLOTER21
     *                B(120),BETA,HW(120),T2,DRAG,W,XD,T,XP,M,IT,       PLOTER22
     *                DELTAS,TX,EST(120),C,RO,KA,MMAX(120),TEST(120),   PLOTER23
     *               N(120),PHALF                                       PLOTER24
      COMMON/OUT/NPRINT,NPLOT,END                                       PLOTER25
C                                                                       PLOTER26
C * * * * * * * * *  SET UP VALUES FOR PLOT AND CREATE PLOT             PLOTER27
```

```
      NFUN=2                                                            PLOTER28
C * * * * * * * * SET UP MIN AND MAX LIMITS FOR PLOT                    PLOTER29
      FMIN(1)=FX(1,1)                                                   PLOTER30
      FMIN(2)=FX(2,1)                                                   PLOTER31
      FMAX(1)=FX(1,1)                                                   PLOTER32
      FMAX(2)=FX(2,1)                                                   PLOTER33
C * * * * * * * * SET UP MIN AND MAX LIMIMTS FOR PITCH AND HEAVE RATIO  PLOTER34
      FMNP=FX(2,NWAVE)                                                  PLOTER35
      FMXP=FX(2,NWAVE)                                                  PLOTER36
      FMNH=FX(1,NWAVE)                                                  PLOTER37
      FMXH=FX(1,NWAVE)                                                  PLOTER38
C                                                                       PLOTER39
      DO 200 I=1,IB                                                     PLOTER40
      IF(FX(1,I).LT.FMIN(1))FMIN(1)=FX(1,I)                             PLOTER41
      IF(FX(1,I).GT.FMAX(1))FMAX(1)=FX(1,I)                             PLOTER42
      IF(FX(2,I).LT.FMIN(2))FMIN(2)=FX(2,I)                             PLOTER43
      IF(FX(2,I).GT.FMAX(2))FMAX(2)=FX(2,I)                             PLOTER44
      IF(I.LE.NWAVE)GO TO 200                                           PLOTER45
      IF(FX(1,I).LT.FMNH)FMNH=FX(1,I)                                   PLOTER46
      IF(FX(1,I).GT.FMXH)FMXH=FX(1,I)                                   PLOTER47
      IF(FX(2,I).LT.FMNP)FMNP=FX(2,I)                                   PLOTER48
      IF(FX(2,I).GT.FMXP)FMXP=FX(2,I)                                   PLOTER49
  200 CONTINUE                                                          PLOTER50
      IF(IPT.EQ.0) GO TO 800                                            PLOTER51
C * * * * * * * COMPUTE RATIOES                                         PLOTER52
      COL3 = (FMXH-FMNH)/(2.*RO)                                        PLOTER53
      COL4 = (FMXP-FMNP)/((4.*PI*RO)/LAMBDA)                            PLOTER54
      WRITE(4,700) COL3,COL4                                            PLOTER55
  700 FORMAT(1H1,"  HEAVE AMPLITUDE/WAVEHEIGHT = ",E12.6,/,2X,          PLOTER56
     *      "  PITCH AMPLITUDE/(2.*PI*WAVEHEIGHT/LAMBDA)  = ",E12.6)    PLOTER57
C                                                                       PLOTER58
  800 CONTINUE                                                          PLOTER59
      NVAR(1)=10H HEAVE                                                 PLOTER60
      NVAR(2)=10H PITCH                                                 PLOTER61
      N1=2                                                              PLOTER62
      X0=XA                                                             PLOTER63
      DELX = HMAX                                                       PLOTER64
      IF(NPLOT.EQ.1)CALL PLOT2(FX,FMIN,FMAX,NVAR,NFUN,N1,IB,X0,DELX)    PLOTER65
      RETURN                                                            PLOTER66
      END                                                               PLOTER67
      SUBROUTINE TRAP(F,DX,NPTS,ANS)                                    TRAP   2
C                                                                       TRAP   3
C   INPUT:                                                              TRAP   4
C        F          ARRAY OF FUNCTIONAL VALUES OF THE INTEGRAND         TRAP   5
C        DX         THE X INTERVAL BETWEEN VALUES                       TRAP   6
C        NPTS       THE NUMBER OF VALUES GIVEN                          TRAP   7
C   OUTPUT:                                                             TRAP   8
C        ANS        THE VALUE OF THE INTEGRAL                           TRAP   9
C                                                                       TRAP  10
      DIMENSION F(NPTS)                                                 TRAP  11
      ANS=0.0                                                           TRAP  12
      IF(NPTS.LT.2)GO TO 999                                            TRAP  13
      DO 1 I=1,NPTS                                                     TRAP  14
    1 ANS=ANS+F(I)                                                      TRAP  15
      ANS=DX*(ANS-0.5*(F(1)+F(NPTS)))                                   TRAP  16
  999 CONTINUE                                                          TRAP  17
      RETURN                                                            TRAP  18
      END                                                               TRAP  19
      FUNCTION RMP(T,START,RISE)                                        RMP    2
```

```
C * * * * * * * *  THIS FUNCTION IS USED TO GRADUALLY IMPLIMENT THE WAVE
C
C        T         CURRENT TIME
C        START     TIME TO START RAMP FROM 0.0 TO 1.0
C        RISE      THE LENGTH OF THE RISE FROM 0.0 TO 1.0
C
      H=0.0
      IF(T.LT.START)GO TO 99
      IF(RISE.EQ.0.0)GO TO 80
      TOP=T-START
      H=1.0
      IF(TOP.LT.RISE)H=TOP/RISE
      GO TO 99
   80 H=1.
      IF(T.EQ.START)H=0.5
   99 RMP=H
      RETURN
      END
```

LISTING OF COMPUTER PROGRAM FOR CALCOMP PLOTS

```
      PROGRAM PLTHSP(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE7,TAPE9)
      ITAPE = 7
      CALL CALPLT(ITAPE)
      STOP
      END
      SUBROUTINE CALPLT(ITAPE)
      DIMENSION TIME(4003),PITCH(4003),HEAVE(4003)
     *          ,IBUF(1000),BWACL(4003),CGACL(4003)
      LOGICAL ACCEL
C
C        CAL COMP PLOT OF PITCH AND HEAVE VERSUS TIME
C
      IREAD = 5
      READ(IREAD,10) XAXIS,YAXISP,YAXISH,HT
   10 FORMAT(8F10.0)
      ACCEL = .FALSE.
      READ(IREAD,20) IA
   20 FORMAT(I10)
      IF(IA.EQ.1) ACCEL = .TRUE.
      IF(ACCEL) READ(IREAD,10) YAXISB,YAXISC
      CALL READT(TIME,HEAVE,PITCH,BWACL,CGACL,NPTS)
      CALL PLOTS(IBUF,1000,7)
      CALL PLOT(0.5,1.0,-3)
      CALL ESCALE(TIME,XAXIS,NPTS,1)
      CALL ESCALE(HEAVE,YAXISH,NPTS,1)
      CALL ESCALE(PITCH,YAXISP,NPTS,1)
      IF(ACCEL) CALL ESCALE(BWACL,YAXISB,NPTS,1)
      IF(ACCEL) CALL ESCALE(CGACL,YAXISC,NPTS,1)
      N1 = NPTS+1
      N2 = NPTS+2
      N3 = NPTS+3
      CALL EAXIS(0.0,0.0,15HTIME IN SECONDS,-15,XAXIS,0.0,
     *          TIME(N1),TIME(N2),TIME(N3),HT)
      CALL EAXIS(0.0,0.0,13HHEAVE IN FEET,13,YAXISH,90.0,
     *    HEAVE(N1),HEAVE(N2),HEAVE(N3),HT)
        TEMP = TIME(N2)
        TIME(N2) = TIME(N2)/TIME(N3)
        HEAVE(N2) = HEAVE(N2)/HEAVE(N3)
      CALL LINE(TIME,HEAVE,NPTS,1,0,0)
      TIME(N2) = TEMP
      XNEW = XAXIS+3.
      YNEW = 1.0
      CALL PLOT(XNEW,0.0,-3)
      CALL EAXIS(0.0,0.0,15HTIME IN SECONDS,-15,XAXIS,0.0,
     *       TIME(N1),TIME(N2),TIME(N3),HT)
      CALL EAXIS(0.0,0.0,13HPITCH IN RAD.,13,YAXISP,90.0,
     *     PITCH(N1),PITCH(N2),PITCH(N3),HT)
      TIME(N2) = TIME(N2)/TIME(N3)
      PITCH(N2) = PITCH(N2)/PITCH(N3)
      CALL LINE(TIME,PITCH,NPTS,1,0,0)
      IF(.NOT.ACCEL) GO TO 30
        TIME(N2) = TEMP
      CALL PLOT(XNEW,0.0,-3)
        CALL EAXIS(0.0,0.0,15HTIME IN SECONDS,-15,XAXIS,0.0,TIME(N1),
     *            TIME(N2),TIME(N3),HT)
        CALL EAXIS(0.0,0.0,16HBOW ACCELERATION,16,YAXISB,90.0,BWACL(N1),
     *             BWACL(N2),BWACL(N3),HT)
        TIME(N2) = TIME(N2)/TIME(N3)
        BWACL(N2) = BWACL(N2)/BWACL(N3)
```

```
         CALL LINE(TIME,BWACL,NPTS,1,0,0)                                 CALP  56
C                                                                         CALP  57
         TIME(N2) = TEMP                                                  CALP  58
      CALL PLOT(XNEW,0.0,-3)                                              CALP  59
         CALL EAXIS(0.0,0.0,15HTIME IN SECONDS,-15,XAXIS,0.0,TIME(N1),    CALP  60
     *                 TIME(N2),TIME(N3),HT)                              CALP  61
         CALL EAXIS(0.0,0.0,15HCG ACCELERATION,15,YAXISC,90.0,CGACL(N1),  CALP  62
     *                CGACL(N2),CGACL(N3),HT)                             CALP  63
         TIME(N2) = TIME(N2)/TIME(N3)                                     CALP  64
         CGACL(N2) = CGACL(N2)/CGACL(N3)                                  CALP  65
         CALL LINE(TIME,CGACL,NPTS,1,0,0)                                 CALP  66
   30 CONTINUE                                                            CALP  67
      CALL PLOT(30.0,0.0,999)                                             CALP  68
      RETURN                                                              CALP  69
      END                                                                 CALP  70
      SUBROUTINE READT(TIME,HEAVE,PITCH,BWACL,CGACL,NPTS)                 READ   2
      DIMENSION X(6),HEAVE(1),PITCH(1)                                    READ   3
     *          ,TIME(1),BWACL(1),CGACL(1)                                READ   4
      I = 0                                                               READ   5
    5 CONTINUE                                                            READ   6
      I = I+1                                                             READ   7
      READ(9) TIME(I),(X(I),I=4,6),BWACL(I),CGACL(I)                      READ   8
      IF(EOF(9))10,15                                                     READ   9
   15 CONTINUE                                                            READ  10
      WRITE(6,20) TIME(I),(X(J),J=4,6),BWACL(I),CGACL(I)                  READ  11
   20 FORMAT(1H ,6(F7.2,2X))                                              READ  12
      HEAVE(I) = X(5)                                                     READ  13
      PITCH(I) = X(6)                                                     READ  14
      IF(I.GE.4000) GO TO 10                                              READ  15
      GO TO 5                                                             READ  16
   10 CONTINUE                                                            READ  17
      NPTS = I-1                                                          READ  18
      RETURN                                                              READ  19
      END                                                                 READ  20
      SUBROUTINE EAXIS(XPAGE,YPAGE,IBCD,NCHAR,AXLEN,ANGLE,FIRSTV,         EAXIS  2
     *                     DELTAV,DELTAU,HT)                              EAXIS  3
      DIMENSION IBCD(1)                                                   EAXIS  4
C                                                                         EAXIS  5
C        THIS ROUTINE WORKS LIKE THE CALCOMP AXIS WITH THE                EAXIS  6
C        EXCEPTION THAT THE TICK MARKS ARE NOT NECCESSARILY               EAXIS  7
C        EVERY INCH AND THE HEIGHT OF THE CHARACTERS IS INPUTTED          EAXIS  8
C                                                                         EAXIS  9
      CALL PLOT(XPAGE,YPAGE,3)                                            EAXIS 10
      ISN = ISIGN(1,NCHAR)                                                EAXIS 11
         ISGN = SIGN(1.,DELTAV)                                           EAXIS 12
           AMIN = FIRSTV                                                  EAXIS 13
           X    = XPAGE                                                   EAXIS 14
           Y    = YPAGE                                                   EAXIS 15
           XNUM = FIRSTV-DELTAV                                           EAXIS 16
           N    = AXLEN/DELTAU                                            EAXIS 17
           IF(N*DELTAU.LT.AXLEN) N=N+1                                    EAXIS 18
             AMAX = AMIN+(N*DELTAV)                                       EAXIS 19
           NDIG = NDIGIT(AMIN,AMAX,DELTAU,ND)                             EAXIS 20
   10 CONTINUE                                                            EAXIS 21
         TEST = (NDIG*HT) + HT                                            EAXIS 22
         IF(TEST.GT.DELTAU) HT=HT/2.                                      EAXIS 23
         IF(TEST.GT.DELTAU) GO TO 10                                      EAXIS 24
           AYN  = (1.5*HT)                                                EAXIS 25
           BYN = (((NDIG-2)*HT)/2.+.5*HT)                                 EAXIS 26
```

```
        N = N+1                                                          EAXIS 27
        TANG = (90.+ANGLE)/57.2958                                       EAXIS 28
        ANG  = ANGLE/57.2958                                             EAXIS 29
        ST   = SIN(TANG)                                                 EAXIS 30
        CT   = COS(TANG)                                                 EAXIS 31
        S    = SIN(ANG)                                                  EAXIS 32
        C    = COS(ANG)                                                  EAXIS 33
      DO 30 I=1,N                                                        EAXIS 34
        IF(I.EQ.1) GO TO 20                                              EAXIS 35
        X = X+DELTAU*C                                                   EAXIS 36
        Y = Y+DELTAU*S                                                   EAXIS 37
        CALL PLOT(X,Y,2)                                                 EAXIS 38
      IF(I.EQ.N) GO TO 20                                                EAXIS 39
        XT = X+(.1*CT*ISN)                                               EAXIS 40
        YT = Y+(.1*ST*ISN)                                               EAXIS 41
        CALL PLOT(XT,YT,2)                                               EAXIS 42
   20 XN = X+AYN*CT*ISN-BYN*C                                            EAXIS 43
      YN = Y+AYN*ST*ISN-BYN*S                                            EAXIS 44
      XNUM = XNUM+DELTAV                                                 EAXIS 45
        CALL NUMBER(XN,YN,HT,XNUM,ANGLE,ND)                              EAXIS 46
        CALL PLOT(X,Y,3)                                                 EAXIS 47
   30 CONTINUE                                                           EAXIS 48
      XSP = (((AXLEN/HT)/2.)-(IABS(NCHAR)/2.))*HT                        EAXIS 49
      YSP = 3.5*HT                                                       EAXIS 50
        XT = XPAGE + XSP*C + ISN*YSP*CT                                  EAXIS 51
        YT = YPAGE + XSP*S + ISN*YSP*ST                                  EAXIS 52
        CALL SYMBOL(XT,YT,HT,IBCD,ANGLE,IABS(NCHAR))                     EAXIS 53
      RETURN                                                             EAXIS 54
      END                                                                EAXIS 55
      FUNCTION NDIGIT(AMIN,AMAX,ANUM,ND)                                 NDIG   2
C                                                                        NDIG   3
C          FINDS THE NUMBER OF DIGITS NECCESSARY TO PRINT                NDIG   4
C          EVEN INCREMENT OF THE FUNCTION ON THE AXIS                    NDIG   5
C                                                                        NDIG   6
C          NDIGIT    THE NUMBER OF PLACES IN THE ENTIRE NUMBER           NDIG   7
C          ND        THE NUMBER OF DECIMAL PLACES                        NDIG   8
C          ANUM      THE VALUE GIVEN TO EACH INCREMENT ON THE AXIS       NDIG   9
C                                                                        NDIG  10
      IF(ABS(AMIN).LT.ABS(AMAX)) GO TO 20                                NDIG  11
      IF(ABS(AMIN).EQ.ABS(AMAX).AND.AMAX.NE.0) GO TO 20                  NDIG  12
      IF(ABS(AMIN).GT.ABS(AMAX)) GO TO 10                                NDIG  13
        AMAX = 1.                                                        NDIG  14
        AMIN = -1.                                                       NDIG  15
        GO TO 20                                                         NDIG  16
   10      AMAX = ABS(AMIN)                                              NDIG  17
   20 IF(AMAX.LE.1.) GO TO 50                                            NDIG  18
        NDIV = 10                                                        NDIG  19
        I    = 1                                                         NDIG  20
   30      IF(AMAX/NDIV.LT.1.) GO TO 40                                  NDIG  21
           I = I+1                                                       NDIG  22
           NDIV = NDIV*10                                                NDIG  23
           GO TO 30                                                      NDIG  24
   40   NDIGIT = I+3                                                     NDIG  25
      ND = 2                                                             NDIG  26
      GO TO 80                                                           NDIG  27
   50 NDIV = 10                                                          NDIG  28
      I    = 1                                                           NDIG  29
   60   IF(AMAX*NDIV.GT.1.) GO TO 70                                     NDIG  30
        I = I+1                                                          NDIG  31
```

```
      NDIV = NDIV*10                                                    NDIG  32
      GO TO 60                                                          NDIG  33
   70 NDIGIT = I+2                                                      NDIG  34
      ND = I                                                            NDIG  35
   80 DD = FLOAT(ND)                                                    NDIG  36
      X = ANUM*(10**DD)                                                 NDIG  37
      IX = X                                                            NDIG  38
      IF(X-FLOAT(IX).LT..0001) GO TO 90                                 NDIG  39
      DD = DD+1                                                         NDIG  40
      ND = ND+1                                                         NDIG  41
      NDIGIT = NDIGIT+1                                                 NDIG  42
      GO TO 80                                                          NDIG  43
   90 CONTINUE                                                          NDIG  44
      RETURN                                                            NDIG  45
      END                                                               NDIG  46
      SUBROUTINE ESCALE(ARRAY,AXLEN,NPTS,INC)                           ESCAL  2
C                                                                       ESCAL  3
C        FINDS THE SCALE TO BE USED ON THE AXIS -                       ESCAL  4
C        ARRAY MUST HAS THREE UNUSED POSITIONS                          ESCAL  5
C              ARRAY(NPTS+1) = FIRSTV                                   ESCAL  6
C              ARRAY(NPTS+2) = DELTAV  (THE INCREMENT BETWEEN TICK MARKS ESCAL  7
C                                       VALUES - NUMBERS)               ESCAL  8
C              ARRAY(NPTS+3) = DELTAU (THE INCREMENT IN INCHES          ESCAL  9
C                                  BETWEEN TICK MARKS )                 ESCAL 10
C                                                                       ESCAL 11
C                                                                       ESCAL 12
      DIMENSION ARRAY(1)                                                ESCAL 13
      AMIN = ARRAY(1)                                                   ESCAL 14
      AMAX = ARRAY(1)                                                   ESCAL 15
      ISGN = ISIGN(1,INC)                                               ESCAL 16
      INC = IABS(INC)                                                   ESCAL 17
          DO 10 I=1,NPTS,INC                                            ESCAL 18
             IF(ARRAY(I).LT.AMIN) AMIN=ARRAY(I)                         ESCAL 19
             IF(ARRAY(I).GT.AMAX) AMAX=ARRAY(I)                         ESCAL 20
   10     CONTINUE                                                      ESCAL 21
   20   AUNIT = UNIT(AMIN,AMAX,AXLEN,N,ANUM)                            ESCAL 22
       CALL ADJUST(AMIN,AMAX,AUNIT,AXLEN,N,ANUM)                        ESCAL 23
      ARRAY(NPTS+1) = AMIN                                              ESCAL 24
      ARRAY(NPTS+2) = ANUM*ISGN                                         ESCAL 25
        IF(ISGN.EQ.-1)ARRAY(NPTS+1) = AMAX                              ESCAL 26
      ARRAY(NPTS+3) = AUNIT                                             ESCAL 27
      IF(ABS(ANUM).EQ.AUNIT) ARRAY(NPTS+2) = 1.*ISGN                    ESCAL 28
      IF(ABS(ANUM).EQ.AUNIT) ARRAY(NPTS+3) = 1.                         ESCAL 29
      RETURN                                                            ESCAL 30
      END                                                               ESCAL 31
      SUBROUTINE ADJUST(AMIN,AMAX,AUNIT,AXLEN,N,ANUM)                   JUST   2
C                                                                       JUST   3
C        GIVEN AMIN AND AMAX WHICH ARE DISTINCT VALUES, ADJUST          JUST   4
C        THEM SO THAT THEY ARE EVEN MULTIPLES OF AUNIT                  JUST   5
C                                                                       JUST   6
      K = 1                                                             JUST   7
      MIN = AMIN/ANUM                                                   JUST   8
      IF(AMIN.LT.MIN*ANUM) MIN = MIN-1                                  JUST   9
      AMIN = MIN*ANUM                                                   JUST  10
      MAX = AMAX/ANUM                                                   JUST  11
      IF(AMAX.GT.MAX*ANUM) MAX = MAX+1                                  JUST  12
      AMAX = MAX*ANUM                                                   JUST  13
   10 TERM = AMIN+(N-K)*ANUM                                            JUST  14
      IF(TERM.LT.AMAX) GO TO 20                                         JUST  15
```

```
      K = K+1
      GO TO 10
   20 AUNIT = AXLEN/(N-K+1)
      N = AXLEN/AUNIT+1
      RETURN
      END
      FUNCTION UNIT(AMIN,AMAX,AXLEN,N,ANUM)
C
C          FINDS THE INCREMENT BETWEEN VALUES TO BE USED ON THE
C          AXIS IN AS FAR AS LABELING THE TICK MARKS
C          FINDS THE  NUMBER OF DIVISIONS TO BE MADE ON  THE AXIS
C          FINDS THE SIZE IN INCHES OF THESE DIVISIONS
C
      IF(AMIN.NE.AMAX) GO TO 10
        AMIN = AMIN-1
        AMAX = AMAX+1
   10 IF(AMAX.LT.1.AND.AMIN.GT.-1)GO TO 110
   30  MIN = AMIN
       MAX = AMAX
        IF(AMAX.GT.MAX) MAX=MAX+1
      IF(AMIN.LT.MIN) MIN=MIN-1
      IF(MIN.LT.0) NWID = MAX+IABS(MIN)
      IF(MIN.GE.0) NWID = MAX-MIN
         NUM = 10
   40  IF(NWID.LT.NUM) GO TO 60
        NUM = NUM*10
        GO TO 40
   60  N = NWID/(NUM/10)
      IF(MIN.LT.0.AND.MAX.GT.0) GO TO 70
       IF(N*(NUM/10).LT.NWID) N=N+1
         ANUM = NUM/10.
         AUNIT = AXLEN/N
         GO TO 160
   70 NN = IABS(MIN)/(NUM/10)
      IF(NN*(NUM/10).LT.IABS(MIN)) NN = NN+1
      N = MAX/(NUM/10)
      IF(N*(NUM/10).LT.MAX) N = N+1
      N = N+NN
      ANUM = NUM/10.
      AUNIT = AXLEN/N
      GO TO 160
  110 NUM=10
  120 IF(AMAX*NUM.GT.1) GO TO 130
      NUM = NUM*10
      GO TO 120
  130   UNITT = 1./NUM
  140   N1     = AMIN*NUM
        N2     = AMAX*NUM
          IF(AMIN*NUM.LT.N1) N1=N1-1
          IF(AMAX*NUM.GT.N2) N2=N2+1
          IF(N1.NE.N2) GO TO 150
            AMIN = AMIN-UNITT
            AMAX = AMAX-UNITT
            GO TO 140
  150     N     = N2-N1
          ANUM = UNITT
          IF(AMIN.LT.0.AND.AMAX.LT.0) N=N1-N2
          IF(AMIN.LT.0.AND.AMAX.GE.0) N=N2-N1
          AUNIT = AXLEN/N
```

```
160   IF(N.GT.5) GO TO 170                                              UNIT  55
      N = N*2                                                           UNIT  56
      ANUM = ANUM/2.                                                    UNIT  57
      AUNIT = AUNIT/2.                                                  UNIT  58
      GO TO 160                                                         UNIT  59
170 UNIT = AUNIT                                                        UNIT  60
    RETURN                                                              UNIT  61
    END                                                                 UNIT  62
```

INITIAL DISTRIBUTION

Copies	
1	WES
1	CHONR/438 Cooper
2	NRL
	1 Code 2027
	1 Code 2829
1	ONR/Boston
1	ONR/Chicago
1	ONR/Pasadena
1	NORDA
4	USNA
	1 Tech Lib
	1 Nav Sys Eng Dept
	1 B. Johnson
	1 Bhattacheryya
3	NAVPGSCOL
	1 Library
	1 T. Sarpkaya
	1 J. Miller
1	NADC
3	NOSC
	1 Library
	1 Fabula
	1 Hoyt
1	NCSL/712 D. Humphreys
1	NCEL/Code L31
1	NSWC, Dahlgren
1	NUSC/Lib
7	NAVSEA
	1 SEA 0322
	1 SEA 033
	1 SEA 03512/Pierce
	1 SEA 037
	3 SEA 09G32
1	NAVFAC/Code 032C
1	NAVSHIPYD PTSMH/Lib
1	NAVSHIPYD PHILA/Lib
1	NAVSHIPYD NORVA/Lib
1	NAVSHIPYD CHASN/Lib
1	NAVSHIPYD LBEACH/Lib

Copies	
2	NAVSHIPYD MARE
	1 Library
	1 Code 250
1	NAVSHIPYD BREM/Lib
1	NAVSHIPYD PEARL/Code 202.32
8	NAVSEC
	1 SEC 6034B
	1 SEC 6110
	1 SEC 6114H
	1 SEC 6120
	1 SEC 6136
	1 SEC 6140B
	1 SEC 6144G
	1 SEC 6148
1	NAVSEC, NORVA/6660.03 Blount
12	DDC
1	AFOSR/NAM
1	AFFOL/FYS, J. Olsen
1	NSF/Eng Lib
1	LC/Sci and Tech
1	DOT/Lib TAD-491.1
1	MMA, Library
1	U. of BRIDGEPORT/E. Uram
4	U. of CAL/Dept Naval Arch, Berkeley
	1 Library
	1 Webster
	1 Paulling
	1 Wehausen
2	U. of CAL, San Diego
	1 A.T. Ellis
	1 Scripps Inst Lib
3	CIT
	1 Aero Lib
	1 T.Y. Wu
	1 A. Acosta
1	CATHOLIC U. of AMER/CIVIL MECH ENG
1	COLORADO STATE U./ENG RES CEN
1	U. of CONNECTICUT/Scottron
1	CORNELL U./Sears

Copies	
2	FLORIDA ATLANTIC U.
	1 Tech Lib
	1 S. Dunne
2	HARVARD U.
	1 G. Carrier
	1 Gordon McKay Lib
1	U. of HAWAII/Bretschneider
1	U. of ILLINOIS/J. Robertson
3	U. of IOWA
	1 Library
	1 Landweber
	1 Kennedy
1	JOHN HOPKINS U./Phillips
1	KANSAS STATE U./Nesmith
1	U. of KANSAS/Civil Eng Lib
1	LEHIGH U./Fritz Eng Lab Lib
5	MIT
	1 Library
	1 Leehey
	1 Mandel
	1 Abkowitz
	1 Newman
4	U. of MIN/ST. ANTHONY FALLS
	1 Silberman
	1 Schiebe
	1 Wetzel
	1 Song
3	U. of MICH/NAME
	1 Library
	1 Ogilivie
	1 Hammitt
2	U. of NOTRE DAME
	1 Eng Lib
	1 Strandhagen
1	PENN STATE/ARL/B. Parkin
1	PRINCETON U./Mellor
5	SIT
	1 Library
	1 Breslin
	1 Savitsky
	1 P.W. Brown
	1 Fridsma
1	U. of TEXAS/ARL Lib
1	UTAH STATE U./Jeppson

Copies	
2	SOUTHWEST RES INST
	1 Applied Mech Rev
	1 Abramson
2	STANFORD U.
	1 Eng Lib
	1 R. Street
1	STANFORD RES INST/Lib
1	U. of WASHINGTON/ARL Tech Lib
3	WEBB INST
	1 Library
	1 Lewis
	1 Ward
1	WOODS HOLE/Ocean Eng
1	WORCHESTER PI/Tech Lib
1	SNAME/Tech Lib
1	BETHLEHEM STEEL/Sparrows Point
1	BETHLEHEM STEEL/New York/Lib
1	BOLT, BERANEK and NEWMAN/Lib
1	GENERAL DYNAMICS, EB/Boatwright
1	GIBBS and COX/Tech Info
5	HYDRONAUTICS
	1 Library
	1 E. Miller
	1 A. Goodman
	1 V. Johnson
	1 C.C. Hsu
1	LOCKHEED, Sunnyvale/Waid
2	McDONNELL DOUGLAS, Long Beach
	1 J. Hess
	1 T. Cebeci
1	NEWPORT NEWS SHIPBUILDING/Lib
1	NIELSEN ENG and RES
1	OCEANICS
1	ROCKWELL INTERNATIONAL/B. Ujihara
1	SPERRY RAND/Tech Lib
1	SUN SHIPBUILDING/Chief Naval Arch
1	ROBERT TAGGART
1	TRACOR

CENTER DISTRIBUTION

Copies	Code	Name
1	1500	W.E. Cummins
1	1504	V.J. Monacella
1	1506	M.K. Ochi
1	1507	D. Cieslowski
1	1512	J.B. Hadler
1	1520	R. Wermter
1	1521	P. Pien
1	1524	Y.T. Shen
1	1524	W.C. Lin
1	1532	G. Dobay
1	1532	R. Roddy
1	1540	W.B. Morgan
1	1552	J. McCarthy
1	1552	N. Salvesen
1	1560	G. Hagen
1	1560	N. Hubble
10	1562	M. Martin
1	1564	J. Feldman
1	1568	G. Cox
1	1572	M.D. Ochi
1	1572	C.M. Lee
1	1576	W.E. Smith
10	5214.1	Reports Distribution
1	522.1	Unclassified Library (C)
1	522.2	Unclassified Library (A)

DTNSRDC ISSUES THREE TYPES OF REPORTS

1. DTNSRDC REPORTS, A FORMAL SERIES, CONTAIN INFORMATION OF PERMANENT TECHNICAL VALUE. THEY CARRY A CONSECUTIVE NUMERICAL IDENTIFICATION REGARDLESS OF THEIR CLASSIFICATION OR THE ORIGINATING DEPARTMENT.

2. DEPARTMENTAL REPORTS, A SEMIFORMAL SERIES, CONTAIN INFORMATION OF A PRELIMINARY, TEMPORARY, OR PROPRIETARY NATURE OR OF LIMITED INTEREST OR SIGNIFICANCE. THEY CARRY A DEPARTMENTAL ALPHANUMERICAL IDENTIFICATION.

3. TECHNICAL MEMORANDA, AN INFORMAL SERIES, CONTAIN TECHNICAL DOCUMENTATION OF LIMITED USE AND INTEREST. THEY ARE PRIMARILY WORKING PAPERS INTENDED FOR INTERNAL USE. THEY CARRY AN IDENTIFYING NUMBER WHICH INDICATES THEIR TYPE AND THE NUMERICAL CODE OF THE ORIGINATING DEPARTMENT. ANY DISTRIBUTION OUTSIDE DTNSRDC MUST BE APPROVED BY THE HEAD OF THE ORIGINATING DEPARTMENT ON A CASE-BY-CASE BASIS.