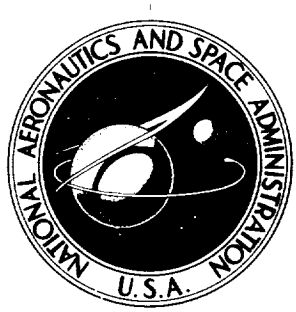
NASA TECHNICAL NOTE

NASA TN D-7347



NASA TN D-7347

DEVELOPMENT AND APPLICATION OF A LOCAL LINEARIZATION ALGORITHM FOR THE INTEGRATION OF QUATERNION RATE EQUATIONS IN REAL-TIME FLIGHT SIMULATION PROBLEMS

by Lawrence E. Barker, Jr., Roland L. Bowles, and Louise H. Williams

Langley Research Center
Hampton, Va. 23665

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • DECEMBER 1973

1. Report No. NASA TN D-7347	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle DEVELOPMENT AND APPLICATION OF A LOCAL LINEARIZATION ALGORITHM FOR THE INTEGRATION OF QUATERNION RATE EQUATIONS IN REAL-TIME FLIGHT SIMULATION PROBLEMS		5. Report Date December 1973 6. Performing Organization Code
7. Author(s) Lawrence E. Barker, Jr.; Roland L. Bowles; and Louise H. Williams, Electronic Associates, Inc.		8. Performing Organization Report No. L-8954 10. Work Unit No. 501-39-11-02
9. Performing Organization Name and Address NASA Langley Research Center Hampton, Va. 23665		11. Contract or Grant No. 13. Type of Report and Period Covered Technical Note
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

High angular rates encountered in real-time flight simulation problems may require a more stable and accurate integration method than the classical methods normally used. A study has been made to develop a general local linearization procedure of integrating dynamic system equations when using a digital computer in real time. The procedure is specifically applied to the integration of the quaternion rate equations. For this application, results are compared to a classical second-order method. The local linearization approach is shown to have desirable stability characteristics and gives significant improvement in accuracy over the classical second-order integration methods.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Local linearization Quaternions Integration	Unclassified – Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	56	Domestic, $3.50 Foreign, $6.00

*For sale by the National Technical Information Service, Springfield, Virginia 22151

CONTENTS

DEVELOPMENT AND APPLICATION OF A LOCAL LINEARIZATION ALGORITHM FOR THE INTEGRATION OF QUATERNION RATE EQUATIONS IN REAL-TIME FLIGHT SIMULATION PROBLEMS

By Lawrence E. Barker, Jr., Roland L. Bowles,
and Louise H. Williams*
Langley Research Center

SUMMARY

High angular rates encountered in real-time flight simulation problems may require a more stable and accurate integration method than the classical methods normally used. A study has been made to develop a general local linearization procedure of integrating dynamic system equations when using a digital computer in real time. The procedure is specifically applied to the integration of the quaternion rate equations. For this application, results are compared to a classical second-order method. The local linearization approach is shown to have desirable stability characteristics and gives significant improvement in accuracy over the classical second-order integration methods.

INTRODUCTION

The quaternion (or Euler parameter) rate equations are widely used for determining the orientation of missiles and aircraft in real-time flight simulation problems. These equations are commonly integrated by using a second-order integration method developed by Adams and Bashforth (AB-2 method). As shown in reference 1, the AB-2 method is sufficiently accurate for simulation studies involving aircraft with moderate values of angular rates; however, with the advent of high performance aircraft and their subsequent simulation, more accurate integration techniques are required.

The purpose of the present study is to further evaluate integration of the quaternion rate equations by the AB-2 method and to develop and evaluate an improved integration scheme to use in real-time simulations. Desirable characteristics for this improved algorithm are: (1) stability and accuracy over a large range of angular rates, (2) one-pass algorithm which could be used at large step sizes, and (3) computer timing and memory requirements comparable to the classical methods commonly used.

---

*Electronic Associates, Inc.

A general algorithm based on a local linearization procedure is developed and is then applied to the digital integration in real time of the quaternion rate equations. The resulting algorithm is referred to as the LL algorithm. The description of the LL implementation on a digital computer is presented in the form of a program flow chart and FORTRAN source listing. Also included is a simplified version of the LL algorithm for those users with limited computer resources.

The LL algorithm is compared with the AB-2 method both analytically and experimentally. This comparison includes stability and error analysis. Included for the experimental study are both analytical inputs and inputs taped from an actual piloted run. In addition, both methods (LL and AB-2) are compared with a variable step seventh-order Runge-Kutta method used as a high-quality approximation to the exact solution.

SYMBOLS

$A', B', C', C_2', C_3', C_i, D', H, G, J, K$ — coefficients in LL quaternion algorithm, where $i = 1, \ldots, 4$

$A(t)$ — time-varying matrix; for quaternions, a skew-symmetric matrix which relates components of a vector to their derivatives due to referencing a rotating reference frame, radians/second

A_p — amplification constant for roll rate p, radians/second

a_i — quaternions (Euler parameters), where $i = 1, \ldots, 4$

B — control matrix of dynamic system

C — transformation matrix relating vehicle body axes to inertial axes

c_{ij} — matrix elements of C, where $i,j = 1, 2, 3$

$e^{A_k h}$ — discrete form of transition matrix

$\overline{F}(\overline{X},t)$ — n-dimensional vector composed of general nonlinear time-varying functions of state vector $\overline{X}$ and time t, per second

$\frac{\partial \overline{F}}{\partial \overline{X}}$ — Jacobian matrix of vector $\overline{F}$ with respect to vector $\overline{X}$, $\left[\frac{\partial \overline{F}}{\partial X_1}, \frac{\partial \overline{F}}{\partial X_2}, \ldots, \frac{\partial \overline{F}}{\partial X_n}\right]$

h	integration interval size, seconds
I	identity matrix
i	imaginary unit, $\sqrt{-1}$
$\bar{K}_i$	constant complex vectors, where $i = 1, 2$
$\bar{K}_i'$	constant real vectors, where $i = 1, 2$
k,n	integer constants
M	matrix which relates quaternions at $t = t_{k+1}$ to those at $t = t_k$
N	norm of $\bar{X}$
$O(h^n) = \phi$	where $\lvert\phi\rvert \leqq K\lvert h^n\rvert$ and $K > 0$
P_k,Q_k	coefficients in general LL algorithm for solution of nonlinear time-varying dynamical system
p,q,r	components of angular velocity vector $\bar{\omega}$, radians/second
t	time, seconds
t_k	value of t at $t = kh$, where $k = 0, 1, 2, \ldots$, seconds
$\bar{u}(t)$	control vector (forcing function of time-varying system)
$\bar{X}(t)$	n-dimensional vector representing system states of a dynamic system
$\bar{X}^N$	$\bar{X}$ whose elements have been normalized
δt	small perturbation in t about t_k
$\delta\bar{X}(t)$	small perturbation in $\bar{X}$ about $\bar{X}_k$
λ	complex number, $\lambda = R(\lambda) + I(\lambda)i$, radians/second

ρ_k parameter in LL quaternion algorithm, $(\omega_k h)/2$, radians

ψ,θ,ϕ Euler angles relating vehicle body axes to inertial axes system, radians

ω magnitude of $\overline{\omega}$, radians/second

ω_c actual computed frequency of quaternions, radians/second

ω_D desired frequency of quaternions, $\omega/2$, radians/second

$\overline{\omega}$ angular velocity vector relative to inertial axes system, radians/second

Where R is any arbitrary scalar or vector variable:

R_k value of R at $t = t_k$

$\dot{R}$ derivative of R with respect to time

R_{AB2} R evaluated by AB-2 method

R_{LL} R evaluated by LL algorithm

ΔR deviation in R from some reference

ϵ_R error in R from some reference

R_{max} maximum value of R

R_o,R(0) initial value of R

Where R is any arbitrary matrix:

R_k value of R at $t = t_k$

R^{-1} inverse of R

R^T transpose of R

ABBREVIATIONS

AB-2 Adams-Bashforth second-order integration

LL local linearization

RK-7 Runge-Kutta seventh-order integration

ANALYSIS

Statement of Problem

An aircraft may be considered a moving coordinate system. From the angular velocity components (p,q,r), an integration of angular rate to determine angular position is done. The quaternion rate equations are normally used in describing this orientation (refs. 2 and 3). In some problems such as the simulation of two aircraft, the relative geometry computations and the integration of the equations of motion, as well as the above mentioned angular rates, require a large number of arithmetic operations per integrating step. These operations are so numerous that an integration method is needed that requires only one evaluation of the derivatives per integration step. The term "time critical" will be used to define problems with this characteristic. A method which does n derivative evaluations per step is defined as an n-pass integration scheme. Higher order multistep methods could be used to increase the accuracy and maintain one evaluation per step, but these methods have other difficulties such as determining the required starting values. At the NASA Langley Research Center an interval size of $h = 1/32$ sec is normally used for real-time flight simulation problems. The integration is typically done with a one-pass integration method such as second-order AB-2. For high angular rates encountered in high-performance aircraft, particularly as subsystems of an air-to-air combat simulation, the computation is not sufficiently accurate using this method. The purpose of the present study is to develop an improved algorithm for the integration of the quaternion rate equations on a digital computer in real time. For this report, angular rates up to 10 rad/sec are considered.

Figure 1 shows how the quaternion equations fit into a simulation problem. Note in this figure the direction cosines c_{ij} are computed algebraically from the quaternions. The angular components p, q, and r of the spin vector $\overline{\omega}$, available from the equations of motion, are taken as inputs to the quaternion block which is treated as the system in this study. Also shown are the initialization and attitude angle readout blocks. The remaining block in the figure is the normalization block. This block is usually added to normalize the quaternions after integration of the quaternion rate equations (appendix A). Any

closed-loop effects through the equations of motion or any visual or other feedback effects due to a pilot are not considered in this report.

The quaternion equations can be written as the following matrix differential equation:

$$\dot{\overline{X}}(t) = A(t)\,\overline{X}(t) \tag{1}$$

where

$$\overline{X}(0) = \overline{X}_0$$

$$A(t) = \frac{1}{2}\begin{bmatrix} 0 & -r & -q & -p \\ r & 0 & -p & q \\ q & p & 0 & -r \\ p & -q & r & 0 \end{bmatrix} \qquad \left(A = -A^T\right) \tag{2}$$

$$\overline{X} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} \tag{3}$$

(Note that a_1, a_2, a_3, and a_4 of this report are equivalent to a_0, a_3, a_2, and a_1, respectively, of ref. 3.) From the eigenvalues of the above system, it is seen that the natural frequency of this system is $\omega/2$, where

$$\omega^2 = p^2 + q^2 + r^2$$

Other methods of calculating attitudes of a rigid body such as direction cosines (refs. 2 and 3) result in a frequency of ω. The reduced frequency is an additional advantage in using the quaternion equations.

For the matrix $A(t)$ defined in equation (2), matrix equation (1) is called the Euler parameter equation where $a_1, \ldots, a_4$ from equation (3) are called Euler parameters or quaternions. The normalization condition for the quaternions is defined as

$$N^2 = a_1^2 + a_2^2 + a_3^2 + a_4^2 = 1 \tag{4}$$

where N is the norm. The term "norm in" will be used to define the system described by equation (1) and satisfying equation (4). When the system is not required to satisfy equation (4), it will be referred to as "norm out." The effect of normalization upon accuracy of computation is discussed later. This report is concerned with the integration of equation (1) in a one-pass system where $A(t)$ defines the quaternion dynamics. The investigation is performed for the classical second-order Adams-Bashforth predictor integration method (AB-2 method) and for the new algorithm (LL algorithm) which was developed to maintain accuracy at high angular rates and at increased step sizes.

AB-2 Method

Before developing the LL algorithm, a look at the solution of equation (1), using the AB-2 method with Euler integration as the starter formula, is in order (ref. 1). The vector AB-2 equation is written

$$\overline{X}_{k+1} = \overline{X}_k + \frac{h}{2}\left(3\dot{\overline{X}}_k - \dot{\overline{X}}_{k-1}\right) \tag{5}$$

Applying equation (5) to equation (1) gives

$$\overline{X}_{k+1} = \overline{X}_k + \frac{h}{2}\left(3A_k\overline{X}_k - A_{k-1}\overline{X}_{k-1}\right) \tag{6}$$

where

$$A_k = A(t_k)$$

Where $\overline{\omega}$ is a constant vector $A = A_k$, by taking the z-transform of equation (6) and solving the resulting characteristic equation as in reference 1, the approximate solution assuming $h\omega << 1$ can be written as

$$\overline{X}(t) = e^{\frac{4h^3\omega^4}{256}t}\left[\overline{K}_1 e^{i\left(\frac{\omega}{2}+\frac{5}{96}h^2\omega^3\right)t} + \overline{K}_2 e^{-i\left(\frac{\omega}{2}+\frac{5}{96}h^2\omega^3\right)t}\right] \tag{7}$$

where K_1 and K_2 are constant complex vectors that depend on initial conditions. The application of Euler's formula yields the real solution

$$\bar{X}(t) = e^{\frac{4h^3\omega^4}{256}t}\left[\bar{K}_1' \cos\left(\frac{\omega}{2} + \frac{5}{96}h^2\omega^3\right)t + \bar{K}_2' \sin\left(\frac{\omega}{2} + \frac{5}{96}h^2\omega^3\right)t\right] \tag{8}$$

where $\bar{K}_1'$ and $\bar{K}_2'$ are constant real vectors. Using initial conditions for equation (1), equation (8) becomes

$$\bar{X}(t) = e^{\frac{4h^3\omega^4}{256}t}\left[I \cos\left(\frac{\omega}{2} + \frac{5}{96}h^2\omega^3\right)t + \frac{A - \frac{4h^3\omega^4}{256}I}{\frac{\omega}{2} + \frac{5}{96}h^2\omega^3} \sin\left(\frac{\omega}{2} + \frac{5}{96}h^2\omega^3\right)t\right]\bar{X}(0)$$

which can be approximated by

$$\bar{X}(t) = e^{\frac{4h^3\omega^4}{256}t}\left[I \cos\left(\frac{\omega}{2} + \frac{5}{96}h^2\omega^3\right)t + \frac{2A}{\omega} \sin\left(\frac{\omega}{2} + \frac{5}{96}h^2\omega^3\right)t\right]\bar{X}(0) \tag{9}$$

The solution for the norm squared is found from equation (9) to be approximately (ref. 1)

$$N^2 = e^{8\left(\frac{h^3\omega^4}{256}\right)t} \tag{10}$$

This equation will be used later in discussing the error in the norm.

General Local Linearization Algorithm

A general algorithm for the solution of a nonlinear system is now developed by the method of local linearization. This approach is taken to produce a single-step one-pass integration algorithm with the intent of applying it to the quaternion rate equations where classical schemes fail for large rates. As mentioned previously, a one-pass scheme is very important for time-critical problems such as large digital simulations in real time. The algorithm is derived by applying the method of perturbation to the differential equations (not states) and then solving exactly the resulting differential equations after dropping higher order terms in the perturbation.

Let the nth order ordinary differential equation describing a general nonlinear time-varying dynamical system be expressed in state variable form as

$$\dot{\overline{X}}(t) = \overline{F}(\overline{X},t) \tag{11}$$

where $\overline{X}(t)$ is the n-dimensional vector representing the system states and $\overline{F}$ is the n-dimensional vector composed of general nonlinear time-varying functions of the state vector $\overline{X}(t)$ and time t. Let $\delta\overline{X}(t)$ be a small perturbation about $\overline{X}_k$ defined as

$$\delta\overline{X}(t) = \overline{X}(t) - \overline{X}_k$$

where $t_k \leqq t \leqq t_{k+1}$; and let δt be a small perturbation in time about t_k defined as

$$\delta t = t - t_k$$

where $t_k \leqq t \leqq t_{k+1}$. Therefore

$$\delta\dot{\overline{X}} = \dot{\overline{X}} = \overline{F}\left(\overline{X}_k,t_k\right) + \left[\frac{\partial\overline{F}}{\partial\overline{X}}\left(\overline{X}_k,t_k\right)\delta\overline{X}(t) + \frac{\partial\overline{F}}{\partial t}\left(\overline{X}_k,t_k\right)\delta t\right] + \ldots \tag{12}$$

where $\overline{F}(\overline{X},t)$ has been expressed as a Taylor series in $\overline{X}(t)$ and t about the point $\left(\overline{X}_k,t_k\right)$. When equation (12) is integrated, it becomes

$$\delta\overline{X}\left(t_{k+1}\right) = A_k^{-1}\left(e^{A_k h} - I\right)\overline{F}\left(\overline{X}_k,t_k\right) + \left(A_k^{-1}\right)^2\left(e^{A_k h} - I - hA_k\right)\frac{\partial\overline{F}}{\partial t}\left(\overline{X}_k,t_k\right) + O\left(h^3\right)$$

where

$$A_k \equiv A\left(t_k\right) = \frac{\partial\overline{F}}{\partial\overline{X}}\left(\overline{X}_k,t_k\right)$$

$$h = t_{k+1} - t_k$$

or

$$\overline{X}_{k+1} = \overline{X}_k + P_k\overline{F}\left(\overline{X}_k,t_k\right) + Q_k\frac{\partial\overline{F}}{\partial t}\left(\overline{X}_k,t_k\right) + O\left(h^3\right) \tag{13}$$

where

$$P_k = A_k^{-1}\left(e^{A_k h} - I\right)$$

$$Q_k = \left(A_k^{-1}\right)^2\left(e^{A_k h} - I - hA_k\right)$$

Equation (13) is a general algorithm for the solution of the nonlinear problem of equation (11) and is related to the exponential method of reference 4. Note that this is a single-step solution to the nonlinear problem with local truncation error being $O(h^3)$.

Now, assuming

$$\bar{F}(\bar{X},t) = A(t)\,\bar{X}(t) + B\bar{u}(t)$$

(a linear time-varying system with forcing function $\bar{u}$), equation (13) becomes

$$\bar{X}_{k+1} = e^{A_k h}\bar{X}_k + Q_k\dot{A}_k\bar{X}_k + P_k B\bar{u}_k + Q_k B\dot{\bar{u}}_k \tag{14}$$

For $B = 0$ (a time-varying system with no forcing function), equation (14) becomes

$$\bar{X}_{k+1} = e^{A_k h}\bar{X}_k + Q_k\dot{A}_k\bar{X}_k \tag{15}$$

Equation (15) is the particular case that is applied in this report to derive an integration algorithm for the quaternion rate equations.

LL Algorithm

Application of equation (15) to the quaternion differential equations results in the LL algorithm. For the matrix series definition of e^{A_k}, it can be shown that

$$e^{A_k h} = I\cos\frac{\omega_k h}{2} + \frac{2A_k}{\omega_k}\sin\frac{\omega_k h}{2} \tag{16}$$

since

$$A_k A_k = -\frac{\omega_k^2 I}{4} \tag{17}$$

where

$$\omega_k = +\sqrt{p_k^2 + q_k^2 + r_k^2}$$

Substituting equation (16) into equation (15) and using equation (17) results in

$$\overline{X}_{k+1} = C_1\overline{X}_k + C_2A_k\overline{X}_k + C_3\dot{A}_k\overline{X}_k + C_4A_k\dot{A}_k\overline{X}_k \tag{18}$$

where the coefficients are defined by

$$C_1 = \cos \rho_k$$

$$C_2 = \frac{2 \sin \rho_k}{\omega_k}$$

$$C_3 = \frac{4}{\omega_k^2}\left(1 - \cos \rho_k\right)$$

$$C_4 = \frac{4}{\omega_k^2}\left(h - \frac{2}{\omega_k} \sin \rho_k\right)$$

and

$$\rho_k = \frac{\omega_k h}{2}$$

With matrix A defined by equation (2), equation (18) can be written

$$\overline{X}_{k+1} = M_k\overline{X}_k \tag{19}$$

where

$$M_k = \begin{bmatrix} H & G & -J & -K \\ -G & H & -K & J \\ J & K & H & G \\ K & -J & -G & H \end{bmatrix}$$

and where the scalar elements of M_k are

$$H = C_1 + C_4 A' \tag{20}$$

$$G = -C_2' r_k - C_3' \dot{r}_k + C_4 B' \tag{21}$$

$$J = C_2' q_k + C_3' \dot{q}_k - C_4 C' \tag{22}$$

$$K = C_2' p_k + C_3' \dot{p}_k - C_4 D' \tag{23}$$

$$A' = -\frac{p_k \dot{p}_k + q_k \dot{q}_k + r_k \dot{r}_k}{4}$$

$$B' = \frac{p_k \dot{q}_k - q_k \dot{p}_k}{4}$$

$$C' = \frac{\dot{p}_k r_k - p_k \dot{r}_k}{4}$$

$$D' = \frac{q_k \dot{r}_k - \dot{q}_k r_k}{4}$$

$$C_2' = \frac{\sin \rho_k}{\omega_k}$$

$$C_3' = \frac{2}{\omega_k^2}\left(1 - \cos \rho_k\right)$$

Equation (19) constitutes the LL algorithm.

The FORTRAN subroutine (QUAT) incorporating the LL algorithm plus normalization is given in appendix B. As mentioned before, this is a single-step algorithm. (See fig. 2 to compare its implementation with that of the AB-2 method.) Note that in addition to using the spin vector $\overline{\omega}$, this algorithm uses the angular acceleration $\dot{\overline{\omega}}$, which is always available from the equations of motion. This adds to its accuracy. If stringent requirements are necessary, a simplified version of this algorithm requiring less time and memory may be used. See appendix C.

Assuming $\bar{\omega}$ is a constant vector, equation (15) becomes

$$\bar{X}_{k+1} = e^{Ah}\bar{X}_k$$

which is the discrete form of

$$\bar{X}(t) = e^{At}\bar{X}(0) \tag{24}$$

or

$$\bar{X}(t) = \left[I \cos\left(\frac{\omega}{2} t\right) + \frac{2A}{\omega} \sin\left(\frac{\omega}{2} t\right)\right]\bar{X}(0) \tag{25}$$

where equation (16) was substituted into equation (24). In contrast to the AB-2 method, equation (9), equation (25) gives the exact solution for the LL algorithm. Refer to appendix D for further stability characteristics of the LL algorithm.

Computational Considerations

The LL algorithm developed in this report, along with the AB-2 method, has been implemented on a CDC 6600 computer at Langley Research Center and has been programed to operate in a real-time mode. Most runs were made at a step size of $h = 1/32$ sec, which is the normal step size that is used for real-time problems. The implementation of the LL algorithm is simple. The flow chart of figure 3 depicts the flow of the real-time program and shows the placement of the LL quaternion subroutine (subroutine QUAT). It should also be noted that the body rate input variables p, q, and r, and their derivatives $\dot{p}$, $\dot{q}$, and $\dot{r}$, will be stored by subroutine SAVE during the first pass of a multipass integration scheme. For a single-pass scheme this storage may be omitted.

Additional compute time for calculating the exponential matrix function of equation (15) by summing a series or by other numerical methods did not exist for the particular application of this report. The skew-symmetric nature of the quaternion system of equation (1) enabled a closed analytical expression to be written. However, the LL algorithm does have slightly larger core and timing requirements than the AB-2 method. The additional requirements were approximately 88 locations and less than 1 percent of the compute time per step size ($h = 1/32$ sec). These requirements are negligible for the typical real-time flight simulation problems; these requirements are even more insignificant when the accuracy gained and the possibility of significantly increasing the step size are considered.

RESULTS AND DISCUSSION

Experiment

Two methods of computing the quaternions (AB-2 and LL) were incorporated into a real-time program. These methods were evaluated simultaneously and the results were compared with each other and with the results obtained independently from a high quality seventh-order variable-step Runge-Kutta integration routine. Most runs were made at $h = 1/32$ sec. As mentioned previously, the quaternion equations are the system or plant of concern for this experiment and require as inputs the angular velocity components p, q, and r of the spin vector $\overline{\omega}$. (The LL algorithm, in addition, requires the angular acceleration components $\dot{p}$, $\dot{q}$, and $\dot{r}$.) The results will be given for four different input types, progressing from the simpler constant inputs to an actual taped piloted run. These inputs are: (1) constant rates, (2) sinusoidal rates, (3) pulse sinusoidal rates, and (4) piloted inputs.

Figure 1 shows the variables of interest in this study. In addition to the errors in the quaternions, the errors encountered in typical direction cosines and in the Euler angles ψ, θ, and ϕ are given. The error in ϕ is of special interest due to the vehicle roll rate p being the main contributor to the angular velocity vector $\overline{\omega}$. These errors would be important in the overall flight simulation problem since they are sent to the simulator and fed back to the equations of motion through pilot control inputs. Though important, the effects of the errors on the complete simulation problem will not be given in this report. The effect of "norm out" will be given for the constant rate input for both the LL and AB-2 methods for comparison (see ref. 1). Results for the other inputs will be for "norm in" unless stated otherwise.

Constant inputs.- The simpler constant rate inputs were chosen since they permitted an analytical solution (eq. (9)) and allow comparison with the results of reference 1. In addition, this case allows the LL algorithm to be used as a reference since it is exact, neglecting computer round-off errors.

Figure 4(a) shows for the AB-2 method the percent error $\epsilon_N{}^2$, from its desired value of 1.0. (See eq. (4).) This error is plotted as a function of time for the constant values $p = q = r = 1/\sqrt{3}$ rad/sec and $h = 1/16$ sec. Normally the Euler integrating starting formula is used to start the AB-2 method. The starting error has been subtracted as in reference 1. Figure 4(b) shows the norm squared error for $h = 1/32$ sec for $p = 1$, 3, and 5 rad/sec ($q = r = 0$). As shown in the figure for $p = 1$, there is only negligible error after 120 sec, but as p is increased, larger errors are encountered at shorter times. Thus the curves indicate the AB-2 method may no longer be applicable for these larger roll rates. The predicted errors shown were calculated from equation (10). The errors in the norm can be eliminated by the addition of normalization to

the resulting quaternions. As stated previously, the LL algorithm has no error in norm for a constant angular velocity. It might be concluded that the LL algorithm does not require normalization. However, for the general case, ω varying, this normalization is shown to add a few digits of accuracy.

See figures 5(a) and 5(b) for two time-history segments of the quaternion a_1 for a constant roll-rate input, $p = 10$ rad/sec $(q = r = 0)$. The upper time-history plot in each figure represents a_1 resulting from the LL algorithm, and the lower represents a_1 resulting from the AB-2 method. Note, in figure 5(a) the divergence of a_1 for AB-2 which could have been predicted from the results shown in figure 4(b). When normalization is added, figure 5(b) shows the error in norm has been eliminated for AB-2.

Figure 6 shows the error in computed frequency of the quaternions ϵ_{ω_c} for AB-2 plotted against the correct or desired frequency ω_D. This frequency error was invariant of normalization. The solid curve shows the error as predicted by equation (9). The errors for LL were essentially zero and are thus not plotted.

Figure 7 shows for AB-2 the error in roll angle versus time for constant values of roll rate p and $h = 1/32$ sec. Even for values of p where ϵ_ϕ is small, the direction cosines could still have an effect in the inertial velocities or relative geometry. (See fig. 1.)

Figure 8 shows how the errors in the direction cosines vary with time. Since the errors for LL are essentially zero for constant rate inputs, LL is used as a reference in calculating these errors for AB-2. The beat oscillation with period of 62.8 sec is due to the phase shift term given in equation (9). Not all elements C_{ij} (where $i,j = 1, 2, 3$) of the direction cosine matrix C vary since $q = r = 0$ for this case. The last two time histories show typical direction cosines, C_{22} and C_{32}.

Sinusoidal inputs.- The following sinusoidal inputs were chosen to provide a drastic time-varying case, drastic in the sense that ω is approximately 10 rad/sec:

$$p = 10 \sin 0.5t$$

$$q = r = 2 \sin t$$

Figure 9(a) shows typical time histories of the quaternions $a_1, \ldots, a_4$, while figure 9(b) gives the Euler angles, ψ, θ, and ϕ. A seventh-order Runge-Kutta solution (RK-7) was used as an independent check. The continuous curves for the AB-2 method and the LL algorithm are denoted by subscripts AB-2 and LL, respectively. Figure 9(c) shows the differences in solution between AB-2 and LL for the direction cosines. Also shown are two typical direction cosines C_{22} and C_{32}. Figure 9(d) gives the difference

in solution for the Euler angles. Also shown are the angular rate inputs p, q, and r. The Euler angle errors are given in percent of one revolution (360^{o}).

Figures 9(a) and 9(b) are plotted on a scale such that any significant error may not be readily apparent. However, if reference is made to table I, which gives typical errors for t = 58, 59, and 60 sec, significant errors will be noted. (For example, at t = 58 sec the AB-2 method shows an error of approximately 14.5^{o}.) The errors are shown for AB-2 with "norm in" and for LL with both "norm in" and "norm out." This table clearly points out the superiority of the LL algorithm over the AB-2 method.

The extent of the nonorthogonality of the direction cosine matrix resulting from the quaternion integration errors is shown for this time-varying case. The criterion used to evaluate the orthogonality of the direction cosines is to multiply the computed matrix C of direction cosines by its transpose C^{T} (appendix A). For the above inputs, the elements of the matrix CC^{T} is compared with the corresponding elements of a unity matrix. At the end of a 60-second run for AB-2 and "norm out," the product of the computed matrix of direction cosines with its transpose produced a matrix with numbers along the main diagonal that differed from unity by about 8×10^{-1}; for the same run for LL and "norm out" the difference was 2×10^{-2}. The off-diagonal numbers which should have been zero contained only the round-off error of the computer for both methods (4×10^{-15}). At the end of a 60-second run for "norm in," the matrix CC^{T}, neglecting round-off error, was equivalent to the unity matrix for both methods, as would be expected.

Sinusoidal pulse inputs.- Results for two sinusoidal pulse cases were chosen to provide time-varying inputs with characteristics similiar to those of an actual piloted roll maneuver. The roll rate was chosen to give roll in one direction only to show the effect of cumulative error in roll angle.

For the first case, a coning run, where, for positive values only,

$$p = 5 \sin 0.25t$$

with

$$q = 0.25 \cos 12t$$

$$r = 0.25 \sin 12t$$

was considered. Figures 10(a) and 10(b) show the quaternions and Euler angles after t = 60 sec. Figures 10(c) and 10(d) show how the errors increase with time for both the direction cosines and Euler angles. The cumulative error in ϕ can easily be seen. Typical errors are given in table II.

In aircraft studies, since the vehicle roll rate p is the main contributer to the angular velocity vector $\overline{\omega}$, a design chart was developed for the second case, a pure roll maneuver using AB-2, where, for positive values only,

$$p = A_p \sin \omega t$$

with

$$q = r = 0$$

and

$$p_{max} = A_p \leqq 10 \text{ rad/sec}$$

$$\dot{p}_{max} = A_p \omega \leqq 10 \text{ rad/sec}^2$$

Figure 11 is a design chart showing curves of constant error in roll angle ϕ for two consecutive sinusoidal pulses in roll rate p using AB-2. This error is percent full scale where full scale is one revolution ($\phi = 360^o$). For example, given for a particular problem that $\dot{p}_{max} = 3.5$ rad/sec^2 and $p_{max} = 7$ rad/sec, the error in roll angle is approximately 3 percent or 10.8^o. From this chart, one can see that the main contribution to error is large values of the angular rate. Even for small accelerations, very large errors will result for large angular rates. The errors for this chart were obtained using LL as the reference. This was done for programing simplicity since the maximum error for LL for the ranges of p and $\dot{p}$ shown in the chart is 0.1 percent (or 0.36^o) for $p_{max} = 10$ rad/sec; thus LL is an adequate reference. Similiar design charts were developed for the second-order Adams-Moulton and Runge-Kutta integration routines, which, even with their additional pass, did not eliminate the errors at the higher roll rates. This could have been predicted from the stability chart in figure 12.

Piloted run.- The final case used as inputs the angular rates and accelerations from an actual piloted run. This was not a very violent run, having a peak roll rate of only 5 rad/sec for a short duration of time. Figures 13(a) and 13(b), depicting errors in direction cosines and Euler angles for AB-2 for a time segment of 130 sec, show p to be the dominant input, the criterion used in selecting the previous inputs. Typical errors for both AB-2 and LL are given in table III.

Effects of Increasing Step Size

Of great importance in real-time digital flight simulation would be the increase in step size without significant loss in accuracy. For the drastic sinusoidal case, figures 14(a) and 14(b) show the results for $h = 1/16$ sec. Similar results for $h = 1/32$ sec were shown in figures 9(a) and 9(b). Typical errors are given in tables I, II, and III for the three time-varying cases covered in this report. On a design chart (similiar to fig. 11) for the LL algorithm for $h = 1/16$ sec, the maximum error in ϕ was 0.4 percent (1.4°). These results show the LL algorithm may be used with a large step size to significantly decrease the total computing time for a solution and still give results which are an order of magnitude more accurate than the AB-2 method using half the step size. For systems other than the quaternions, the $\overline{F}(\overline{X},t)$ of equation (1) may be a more complex vector function and the matrix exponential function e^{Ah} may not be computed directly as in this report; however, the possibility of using a larger step size could still outweigh the increase in computation time necessary to compute the exponential function.

CONCLUDING REMARKS

A general algorithm for the solution of nonlinear systems is developed by the method of local linearization. Applying this general formula, a one-pass algorithm has been developed for the integration of the quaternion differential equations used in real-time digital simulations for six-degree-of-freedom problems. This algorithm has local truncation error of order of the cube of the step size, making it superior to those classical second-order integration schemes which have error of order of the square of the step size. Its superiority, in terms of both accuracy and stability, has been demonstrated for large angular rates. In fact, it can be used with larger step sizes and still retain accuracy and stability. Being a one-pass algorithm, its application to time-critical flight simulation problems is possible and sometimes necessary.

A FORTRAN subroutine has been written to implement this algorithm and its mechanization is very simple. It had a slight increase in timing and memory requirements over the classical second-order integration method used for comparison in this report; however, this increase was insignificant considering the accuracy gained. If stringent requirements are necessary, a simplified version of this algorithm requiring less time and memory may be used. Where large angular rates are anticipated or runs of long duration are to be made, the algorithm should be mechanized. Even without these conditions, the algorithm is as efficient and much more accurate than the classical methods previously used.

Langley Research Center,
National Aeronautics and Space Administration,
Hampton, Va., September 26, 1973.

APPENDIX A

NORMALIZATION

Numerical integration of the direction cosine rate equations produces errors in the resulting direction cosines which in turn cause the resolved components of a given vector to be nonorthogonal. Normally, constraint equations on the direction cosines are added to improve the orthogonality of the transformation between body and inertial coordinates (refs. 5 to 7). In the present report the quaternion rate equations are integrated and the direction cosines are then computed algebraically from the resulting quaternion states $a_1, \ldots, a_4$. This integration may produce errors in the resultant quaternions which cause the components of the quaternion vector $\overline{X}$ of equation (3) to violate the normality equation (4). The ultimate result of this violation is the nonorthogonality of the transformation matrix mentioned above. When quaternion rate equations are used, steepest descent or other methods are employed to satisfy the normality constraint of equation (4) (refs. 2 and 3). In the present report the quaternions $a_1, \ldots, a_4$ are normalized after integration of the rate equations. The implementation is accomplished by dividing the quaternions by the norm $\left(X^T X\right)^{1/2}$

$$a_i{}^N = \frac{a_i}{\left(X^T X\right)^{1/2}} = \frac{a_i}{N} \qquad (i = 1, \ldots, 4)$$

This mechanization is shown in figure 1. The effectiveness of the constraint can be found by multiplying the computed direction cosine matrix by its transpose. For an orthogonal transformation, a unit matrix is the correct result of this multiplication.

APPENDIX B

LL QUATERNION SUBROUTINES

```
      SUBROUTINE QUAT(I)
C
C**** SUBROUTINE FOR EVALUATION OF THE QUATERNION PARAMETERS
C**** USING LOCAL LINEARIZATION
      COMMON
     X       /INTCOMM/ T          , H          , INT        , NEQ        ,
     X                 ISCHEME    , DERINT(2,23)
     X       /BK02   / AB1(2)     , AB2(2)     , AB3(2)     , AB4(2)
     X       /TQUAT  / PK(2)      , QK(2)      , RK(2)      , PDOTK(2) ,
     X                 QDOTK(2)   , RDOTK(2)
      DIMENSION OMEGAK(2), OMEGAK2(2), RHOK(2), C1(2), C2P(2),C3P(2),
     X          C4(2)    , AK(2)     , BK(2)  , CK(2), DK(2) ,EK(2) ,
     X          HK(2)    , SRHOK(2)  , CRHOK(2)
      REAL JK(2), KK(2)
      EQUIVALENCE(C1,CRHOK)
      DATA OM2ZERO / 1.E-6 /
      OMEGAK2(I) = PK(I)*PK(I) + QK(I)*QK(I) + RK(I)*RK(I)
      IF(OMEGAK2(I).LT.OM2ZERO)    OMEGAK2(I) = OM2ZERO
      OMEGAK(I) = SQRT(OMEGAK2(I))
      RHOK(I)   = H*OMEGAK(I)*.5
      SRHOK(I)  = SIN(RHOK(I))
      CRHOK(I)  = COS(RHOK(I))
      C2P(I)    = SRHOK(I)/OMEGAK(I)
      C3P(I)    = 2.*(1. - CRHOK(I))/OMEGAK2(I)
      C4(I)     = 4.*(H - 2.*C2P(I))/OMEGAK2(I)
      AK(I)     =-.25*(PK(I)*PDOTK(I) + QK(I)*QDOTK(I) + RK(I)*RDOTK(I))
      EK(I)     = .25*(PK(I)*QDOTK(I) - QK(I)*PDOTK(I))
      CK(I)     = .25*(RK(I)*PDOTK(I) - PK(I)*RDOTK(I))
      DK(I)     = .25*(QK(I)*RDOTK(I) - RK(I)*QDOTK(I))
      HK(I)     = C1(I) + C4(I)*AK(I)
      GK(I)     =-C2P(I)*RK(I) - C3P(I)*RDOTK(I) + C4(I)*BK(I)
      JK(I)     = C2P(I)*QK(I) + C3P(I)*QDOTK(I) - C4(I)*CK(I)
      KK(I)     = C2P(I)*PK(I) + C3P(I)*PDOTK(I) - C4(I)*DK(I)
      AT1       = HK(I)*AB1(I)+GK(I)*AB2(I)-JK(I)*AB3(I)-KK(I)*AB4(I)
      AT2       =-GK(I)*AB1(I)+HK(I)*AB2(I)-KK(I)*AB3(I)+JK(I)*AB4(I)
      AT3       = JK(I)*AB1(I)+KK(I)*AB2(I)+HK(I)*AB3(I)+GK(I)*AB4(I)
      AB4(I)    = KK(I)*AB1(I)-JK(I)*AB2(I)-GK(I)*AB3(I)+HK(I)*AB4(I)
      AB1(I)    = AT1
      AB2(I)    = AT2
      AB3(I)    = AT3
C
C***  NORMALIZATION OF QUATERNION PARAMETERS
      CORECT    = 1.0/SQRT(AB1(I)*AB1(I) + AB2(I)*AB2(I) + AB3(I)*AB3(I)
     1                   +  AB4(I)*AB4(I))
      AB1(I)    = CORECT*AB1(I)
      AB2(I)    = CORECT*AB2(I)
      AB3(I)    = CORECT*AB3(I)
      AB4(I)    = CORECT*AB4(I)
      RETURN
      END
```

```
      SUBROUTINE SAVE(I)
C**** SUBROUTINE FOR SAVING FORCING FUNCTION VALUES TO BE USED BY THE
C**** QUATERNION PARAMETER SUBROUTINE. REQUIRED WHEN A MULTI-PASS INTEGRATION
C**** ROUTINE IS USED IN THE MAIN PROGRAM TO INSURE UPDATES ONLY ONCE
C**** PER ITERATION.
      COMMON/BK05/P(2),Q(2),R(2),PDOT(2),QDOT(2),RDOT(2)
      COMMON/TQUAT/PK(2),QK(2),RK(2),PDOTK(2),QDOTK(2),RDOTK(2)
      PK(I)=P(I)                    $PDOTK(I)=PDOT(I)
      QK(I)=Q(I)                    $QDOTK(I)=QDOT(I)
      RK(I)=R(I)                    $RDOTK(I)=RDOT(I)
      RETURN
      END
```

APPENDIX C

A SIMPLIFIED ALGORITHM

If one were to assume a zero-order hold on the p, q, and r inputs to the LL algorithm (that is, p, q, and r assumed constant over one iteration), a simplified version of LL may be obtained. This assumption implies that only the first two terms of equation (18) are used. As a result, the equations for A', B', C', D', C_3', and C_4 would be eliminated. This simplification may be necessary for one with limited computer resources and would supply sufficient accuracy for low angular rates. The maximum error on a design chart for this simplified algorithm (similiar to fig. 11) would be 2.5 percent (or 9°). Typical errors are given in tables I, II, and III. It is important to notice that even this simplified LL algorithm is better than the classical second-order methods referred to in this report.

APPENDIX D

STABILITY CHARACTERISTICS OF LL ALGORITHM

The major computational difficulties associated with solving the system

$$\dot{\overline{X}} = A(t)\,\overline{X}(t) \tag{D1}$$

where

$$\overline{X}(0) = \overline{X}_0$$

$$A = \frac{1}{2}\begin{bmatrix} 0 & -r & -q & -p \\ r & 0 & -p & q \\ q & p & 0 & r \\ p & -q & r & 0 \end{bmatrix} \qquad \left(A = -A^T\right) \tag{D2}$$

$$\overline{X}^T\overline{X} = 1.0$$

using the classical integration methods in real time, lie in the fact that the eigenvalues of the system are outside the stability boundaries of the methods. The eigenvalues are purely imaginary with multiplicity two, that is, $\pm\frac{\omega}{2}i$, $\pm\frac{\omega}{2}i$ where $\omega^2 = p^2 + q^2 + r^2$. The basic problem is illustrated for AB-2 in figure 3 where the stability boundary (ref. 8) is shown relative to the eigenvalue location of equation (1). Clearly, the purely imaginary roots always lie outside the stability boundary of the method ($h > 0$) since the stability boundary only touches the imaginary axis at the origin. Therefore, any truncation or roundoff errors introduced in the solution process will, in general, not damp out; and the integral curve can become strongly divergent, increasing without bound as a function of time. All classical explicit methods which are suitable for real-time simulation have stability boundaries that do not include the imaginary axis except at the origin.

The LL algorithm described previously can be written as

$$\overline{X}_{k+1} = M_k\overline{X}_k \tag{D3}$$

where

$$M_k = H\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} + \begin{bmatrix}0 & G & -J & -K\\ -G & 0 & -K & J\\ J & K & 0 & G\\ K & -J & -G & 0\end{bmatrix} \tag{D4}$$

The method has local truncation error of $O(h^3)$. The scalars H, G, J, and K are as defined in equations (20) to (23). The physical meaning of $\overline{X}$ (ref. 3), indicates bounded solutions of the quaternion rate equations (neutrally stable solutions). Therefore, the eigenvalues of M_k should lie close to the unit circle (ref. 9). Indeed a "perfect" algorithm would necessarily have root locations on the unit circle. The eigenvalues of the matrix M_k can be written as

$$\left.\begin{aligned}\zeta_1 &= H + i\gamma\\ \zeta_2 &= H - i\gamma\\ \zeta_3 &= H + i\gamma\\ \zeta_4 &= H - i\gamma\end{aligned}\right\} \tag{D5}$$

where

$$\gamma = +\sqrt{G^2 + J^2 + K^2}$$

The magnitude of ζ_j $(j = 1, \ldots, 4)$ is given as

$$\left|\zeta_j\right|_k^2 = 1 + \left[\left(\frac{Y}{Z}\right)^2 P(Z) + \left(\frac{Y}{Z}\right) Q(Z)\right]_k \tag{D6}$$

where

$$Y = \dot{\omega}_k h^2 \tag{D7}$$

$$Z = \frac{\omega_k h}{2} \tag{D8}$$

$$P(Z) = \left[\frac{1}{2} + \frac{1}{2Z^2}(1 - \cos Z) - \frac{\sin Z}{Z} + \frac{1}{4}\left(\frac{\sin Z}{Z}\right)^2\right] \tag{D9}$$

$$Q(Z) = \left(\frac{\sin Z}{Z} - \cos Z\right) \tag{D10}$$

$$\omega_k = \frac{1}{\omega_k}\left(p_k\dot{p}_k + q_k\dot{q}_k + r_k\dot{r}_k\right) \tag{D11}$$

The error in $\left|\zeta_j\right|_k^2$ is defined as

$$\left[\left(\frac{Y}{Z}\right)^2 P(Z) + \left(\frac{Y}{Z}\right)Q(Z)\right]_k \tag{D12}$$

Note for $\omega_k \neq 0$ and $\dot{\omega}_k \neq 0$,

$$\lim_{h\to 0}\left[\left(\frac{Y}{Z}\right)^2 P(Z) + \left(\frac{Y}{Z}\right)Q(Z)\right]_k = 0 \tag{D13}$$

that is, the method is consistent. Observe also for $Y \neq 0$,

$$\lim_{\omega\to\infty}\left|\zeta_j\right|_k^2 = 1.0 \tag{D14}$$

when h is fixed and $\dot{\omega} \leq M$ (where M is a positive number), which is a desirable asymptotic property. For sufficiently small h

$$\left|\zeta_j\right|_k^2 = 1 + \frac{\dot{\omega}_k\omega_k h^3}{6} + \frac{1}{16}\dot{\omega}_k{}^2 h^4 + O\left(h^5\right) \tag{D15}$$

with a corresponding error defined by equation (12). For the case where ω is a constant,

$$\left|\zeta_j\right|_k^2 = 1.0 \tag{D16}$$

indicating perfect placement of the system eigenvalues. Corresponding results for AB-2 (ref. 1) and RK-2, two popular methods employed in real-time simulation programing, are, respectively,

$$\left|\zeta\right|^2 = 1 + \frac{h^4\omega^4}{32} + O\left(h^5\right) \tag{D17}$$

$$\left|\zeta\right|^2 = 1 + \frac{h^4\omega^4}{64} + O\left(h^5\right) \tag{D18}$$

For this case (ω is a constant) the superiority of the LL algorithm is obvious since there is a definite bias of root placement for the other two methods. The implication of this observation with regard to numerical stability will now be considered.

Let $\overline{X}\left(t_k\right)$ be the exact solution vector for equation (1) at time t_k and $\overline{X}_k$ be the one-step solution of the LL algorithm, and define the error as $\epsilon_k = \overline{X}_k - \overline{X}\left(t_k\right)$. It can be shown (ref. 10) that error propagation satisfies an equation of the form

$$\epsilon_{k+1} = M_k\epsilon_k + \gamma\left(t_k,h\right) \tag{D19}$$

where $\gamma\left(t_k,h\right)$ is the local truncation error. The behavior of ϵ_k for large n depends mainly on the eigenvalues of M_k. If these eigenvalues have magnitudes which are consistently less than one, then the error norm will remain bounded. If ω is a constant and assuming no computer round-off errors, equation (D19) has the solution $\epsilon_j = 0$ for all j. This conclusion is not true for AB-2 and RK-2 since the magnitude of the eigenvalues are always greater than 1 for $\omega \neq 0$. The analysis of equation (D19) in the general case is more difficult and will not be presented here. Experience has indicated, however, that the LL algorithm exhibits a high degree of stability for relative large step sizes, i.e., $h \geq \frac{1}{32}$ over a wide range of ω and $\dot{\omega}$. A possible explanation of this can be deduced from equation (D15). In aerospace applications the acceleration $\dot{\omega}$ must eventually change sign; therefore the eigenvalues of M_k will not consistently lie outside the unit circle. This error will accelerate when $\left|\zeta_j\right|^2 > 1$ and will eventually damp when $\dot{\omega}$ changes sign.

Another indicator of performance of a specific numerical method with regard to the solution of equation (D1) is the magnitude of the variation of the constant of motion $\overline{X}^T X$. Defining $V = \overline{X}^T \overline{X}$ from equation (D3) and substituting equation (D4) there results in

$$V_{k+1} = \left|\zeta_j\right|_k^2 V_k \tag{D20}$$

It can be shown that V is a Liapunov function (ref. 11) for the system and therefore can be employed to study the stability characteristics of the LL algorithm, i.e., stability is implied by the condition (assuming no rounding errors)

$$\left[\left(\frac{Y}{Z}\right)^2 P(Z) + \frac{Y}{Z} Q(Z)\right]_k \leqq 0 \tag{D21}$$

for all k.

REFERENCES

1. Wilson, John W.; and Steinmetz, George G. (With appendix A by Roland L. Bowles): Analysis of Numerical Integration Techniques for Real-Time Digital Flight Simulation. NASA TN D-4900, 1968.

2. Mitchell, E. E. L.; and Rogers, A. E.: Quaternion Parameters in the Simulation of a Spinning Rigid Body. Simulation, vol. 4, no. 6, June 1965, pp. 390-396.

3. Robinson, Alfred C.: On the Use of Quaternions in Simulation of Rigid-Body Motion. WADC Tech. Rep. 58-17, U.S. Air Force, Dec. 1958. (Available from DDC as AD 234 422.)

4. Pope, David A.: An Exponential Method of Numerical Integration of Ordinary Differential Equations. Commun. ACM, vol. 6, no. 8, Aug. 1963, pp. 491-493.

5. Krasny, Louis M.: The Functional Design of a Special-Purpose Digital Computer for Real-Time Flight Simulation. MRL-TDR-62-39, U.S. Air Force, Apr. 1962, pp. 16-21.

6. Rogers, A. E.; and Connolly, T. W.: Analog Computation in Engineering Design. McGraw-Hill Book Co., Inc. 1960, pp. 415-418.

7. Brown, Robert C.; Brulle, Robert V.; and Giffin, Gerald D.: Six-Degree-of-Freedom Flight-Path Study Generalized Computer Program. Pt. I – Problem Formulation. WADD Tech. Rep. 60-781, Pt. I, U.S. Air Force, May 1961, pp. 195-198.

8. Benyon, P. R.: A Review of Numerical Methods for Digital Simulation. Simulation, vol. 11, no. 5, Nov. 1968, pp. 219-238.

9. Gupta, Someshwar C.; and Hasdorff, Lawrence: Fundamentals of Automatic Control. John Wiley & Sons, Inc., c.1970.

10. Todd, John, ed.: Survey of Numerical Analysis. McGraw-Hill Book Co., Inc., 1962, pp. 323-326.

11. La Salle, Joseph; and Lefschetz, Solomon: Stability by Liapunov's Direct Method. Academic Press, Inc., 1961.

TABLE I.- TYPICAL ERRORS IN THE QUATERNIONS, DIRECTION COSINES, AND ATTITUDE ANGLES FOR SINUSOIDAL INPUTS

[p = 10 sin 0.5t and q = r = 2 sin t]

Method (a)	Norm	Errors in quaternions		Errors in direction cosines		Errors in attitude angles, deg		
		Δa_1	Δa_2	ΔC_{22}	ΔC_{32}	$\Delta\psi$	$\Delta\theta$	$\Delta\phi$
t = 58 sec								
AB-2	In	-0.06986	-0.02731	-0.23045	0.13241	-2.90650	5.72346	-14.65202
LL	Out	-.00261	-.00090	-.00250	-.00379	-.01575	.06807	-.08735
LL	In	-.00046	-.00019	-.00136	.00051	-.01575	.2277	-.08735
LL (simplified)	In	-.02857	-.01235	-.09234	.04516	-1.41934	2.06376	-5.68763
LL (h = 1/16)	In	-.00191	-.00079	-.00570	.00218	-.06681	.09817	-.36531
t = 59 sec								
AB-2	In	0.10321	0.02449	0.05563	-0.19871	-1.32987	-2.00458	-12.23567
LL	Out	-.00027	.00071	.00898	.00224	-.00052	.15083	-.08358
LL	In	.00071	-.0002	.00057	-.00134	-.00052	.00836	-.08358
LL (simplified)	In	.06881	.01744	.04269	-.13096	-.64100	-1.48521	-8.23173
LL (h = 1/16)	In	.00331	.00003	.00262	-.00624	-.00835	.02450	-.38890
t = 60 sec								
AB-2	In	0.02519	0.01482	-0.08091	0.08119	0.27846	-0.63834	-7.13934
LL	Out	.00461	.00043	-.00709	-.00646	.00759	.20100	-.06653
LL	In	.00027	.00020	-.00069	.00060	.00759	-.02980	-.06653
LL (simplified)	In	.03054	.02025	-.09911	.10428	.07469	-.76010	-8.84765
LL (h = 1/16)	In	.00134	.00093	-.00357	.00313	.03528	-.12500	-.33423

[a] h = 1/32 sec unless otherwise noted.

TABLE II.- TYPICAL ERRORS IN QUATERNIONS, DIRECTION COSINES, AND ATTITUDE ANGLES FOR SINUSOIDAL PULSE INPUTS

[p = 5 sin 0.25t (positive values only); q = 0.25 cos 12t and r = 0.25 sin 12t]

Method (a)	Norm	Errors in quaternions		Errors in direction cosines		Errors in attitude angles, deg		
		Δa_1	Δa_2	ΔC_{22}	ΔC_{32}	$\Delta\psi$	$\Delta\theta$	$\Delta\phi$
t = 58 sec								
AB-2	In	-0.02484	-0.00115	0.09590	0.16940	0.17599	-0.05000	-11.17587
LL	Out	.00141	.00019	-.00268	.00093	-.01590	.02552	.01392
LL	In	.00002	.00018	-.00009	-.00022	-.01590	.02338	.01392
LL (simplified)	In	.00706	.00125	-.002769	-.06931	-.18825	-.12289	4.27829
LL (h = 1/16)	In	.00043	.00078	-.00171	-.00391	-.07317	.09166	.24673
t = 59 sec								
AB-2	In	-0.06402	-0.00073	-0.19912	0.04346	-0.08003	0.15304	-11.70024
LL	Out	-.00083	-.00045	-.00007	-.00241	-.05402	-.01411	.01083
LL	In	.00007	-.00043	-.00020	.00001	-.05402	-.01342	.01083
LL (simplified)	In	.02225	-.00161	.06546	-.00529	.10261	-.43160	3.75158
LL (h = 1/16)	In	.00139	-.00180	.00412	-.00032	-.21052	-.07737	.23288
t = 60 sec								
AB-2	In	0.09651	0.00189	0.15015	-0.14546	0.11688	-0.17055	-12.02096
LL	Out	-.00049	-.00009	.00111	.00162	-.00272	.01052	.00821
LL	In	-.00006	-.00009	-.00009	.00009	-.00272	.00990	.00821
LL (simplified)	In	-.02345	-.00021	-.04220	.03122	-.07732	-.01706	3.00933
LL (h = 1/16)	In	-.00173	-.00039	-.00302	.00237	-.01324	.04323	.22151

[a] h = 1/32 sec unless otherwise noted.

TABLE III.- TYPICAL ERRORS IN QUATERNIONS, DIRECTION COSINES, AND ATTITUDE ANGLES FOR TAPED PILOTED RUN

Method (a)	Norm	Errors in quaternions		Errors in direction cosines		Errors in attitude angles, deg		
		Δa_1	Δa_2	ΔC_{22}	ΔC_{32}	$\Delta\psi$	$\Delta\theta$	$\Delta\phi$
t = 108 sec								
AB-2	In	0.00160	0.00681	0.02409	0.01904	0.09140	0.55423	1.69006
LL	Out	.00062	.00028	.00048	.00122	.12874	.09733	.01326
LL	In	.00065	.00020	.00038	-.00109	.12874	.09642	.01326
LL (simplified)	In	.00131	.00040	.00081	-.00209	.25342	.17944	.02594
LL (h = 1/16)	In	.00065	.00019	.00034	-.00116	.13207	.10239	.01130
t = 109 sec								
AB-2	In	0.00253	0.00631	0.02288	0.01922	0.21942	-0.77300	1.54873
LL	Out	.00059	.00016	-.00007	-.00184	.11216	-.08343	-.03234
LL	In	.00061	.00008	-.00018	-.00173	.11216	.08346	-.03234
LL (simplified)	In	.00123	.00012	-.00041	-.00342	.21875	.15583	-.07410
LL (h = 1/16)	In	.00063	.00011	-.00011	-.00173	.11690	.08731	-.02633
t = 110 sec								
AB-2	In	0.00268	0.00470	0.01857	0.01905	0.17777	-0.94058	1.20747
LL	Out	.00035	-.00078	-.00259	-.00344	.00909	-.01809	-.30286
LL	In	.00038	-.00087	-.00271	-.00344	.00909	.01857	-.30286
LL (simplified)	In	.00076	-.00171	-.00523	-.00637	.01344	.02163	-.59196
LL (h = 1/16)	In	.00041	-.00092	-.00290	-.00632	.01283	.02708	-.32036

[a] h = 1/32 sec unless otherwise indicated.

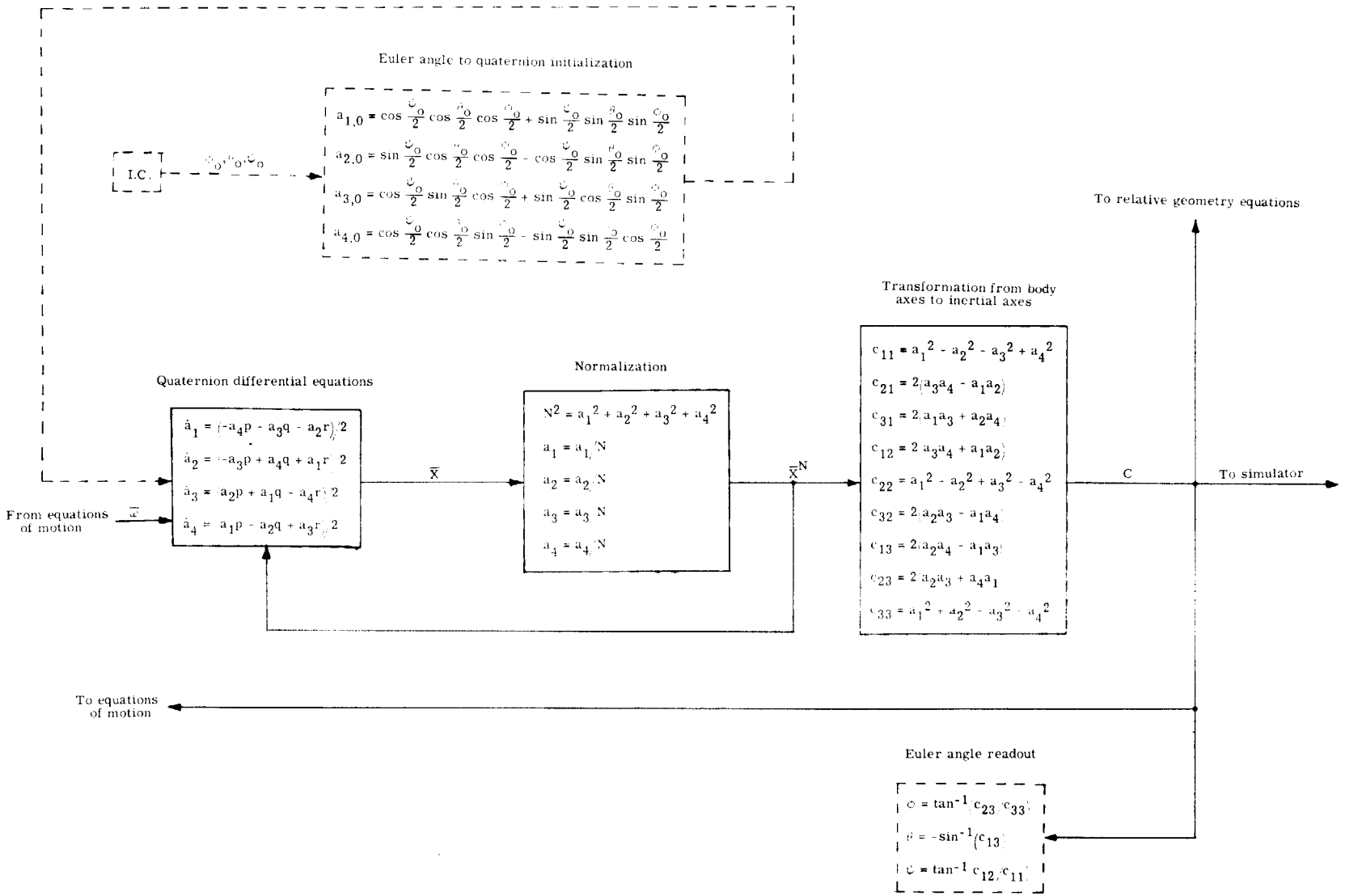


Figure 1.- Block diagram for determining aircraft attitude employing quaternion rate equations.

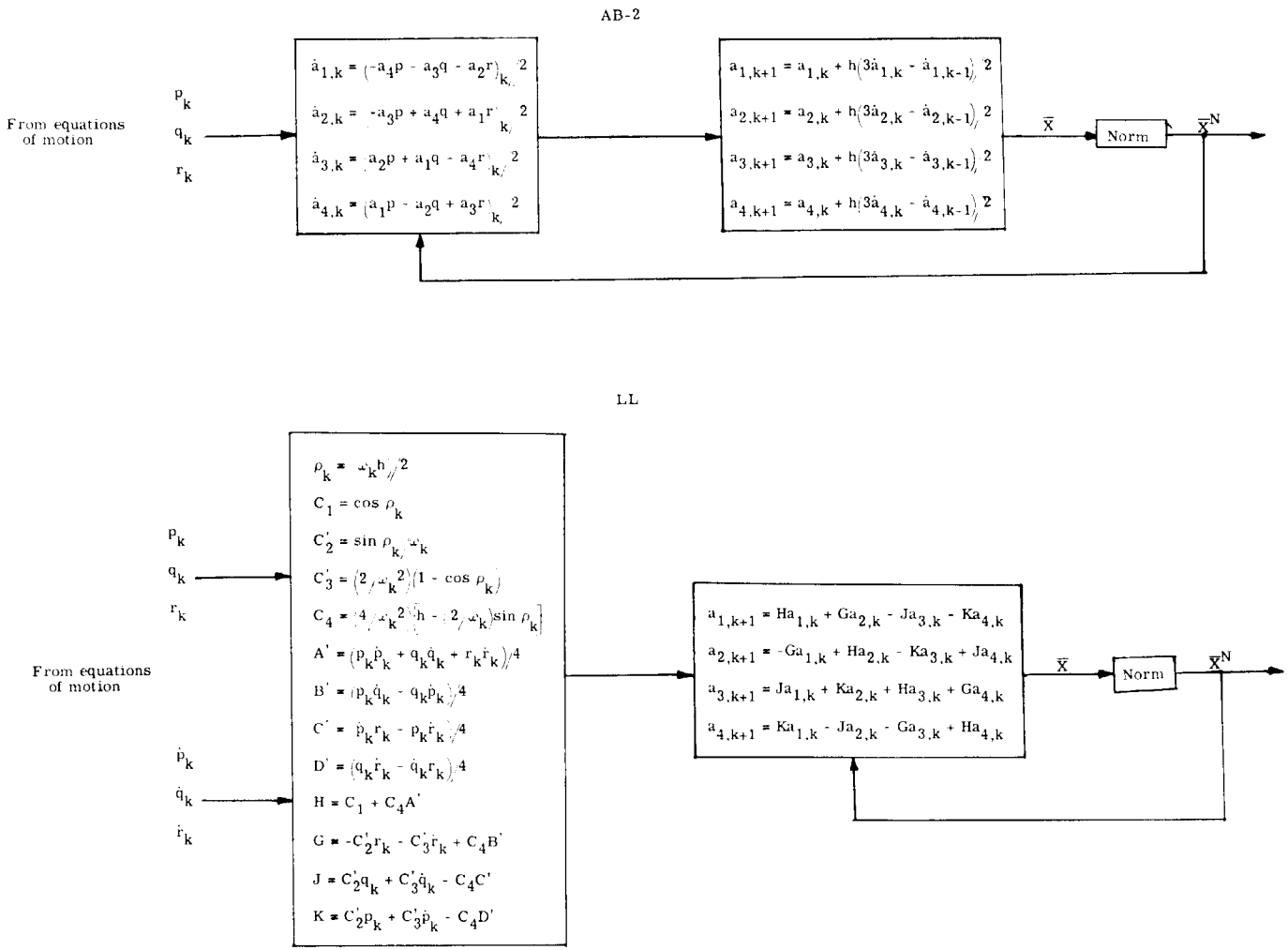
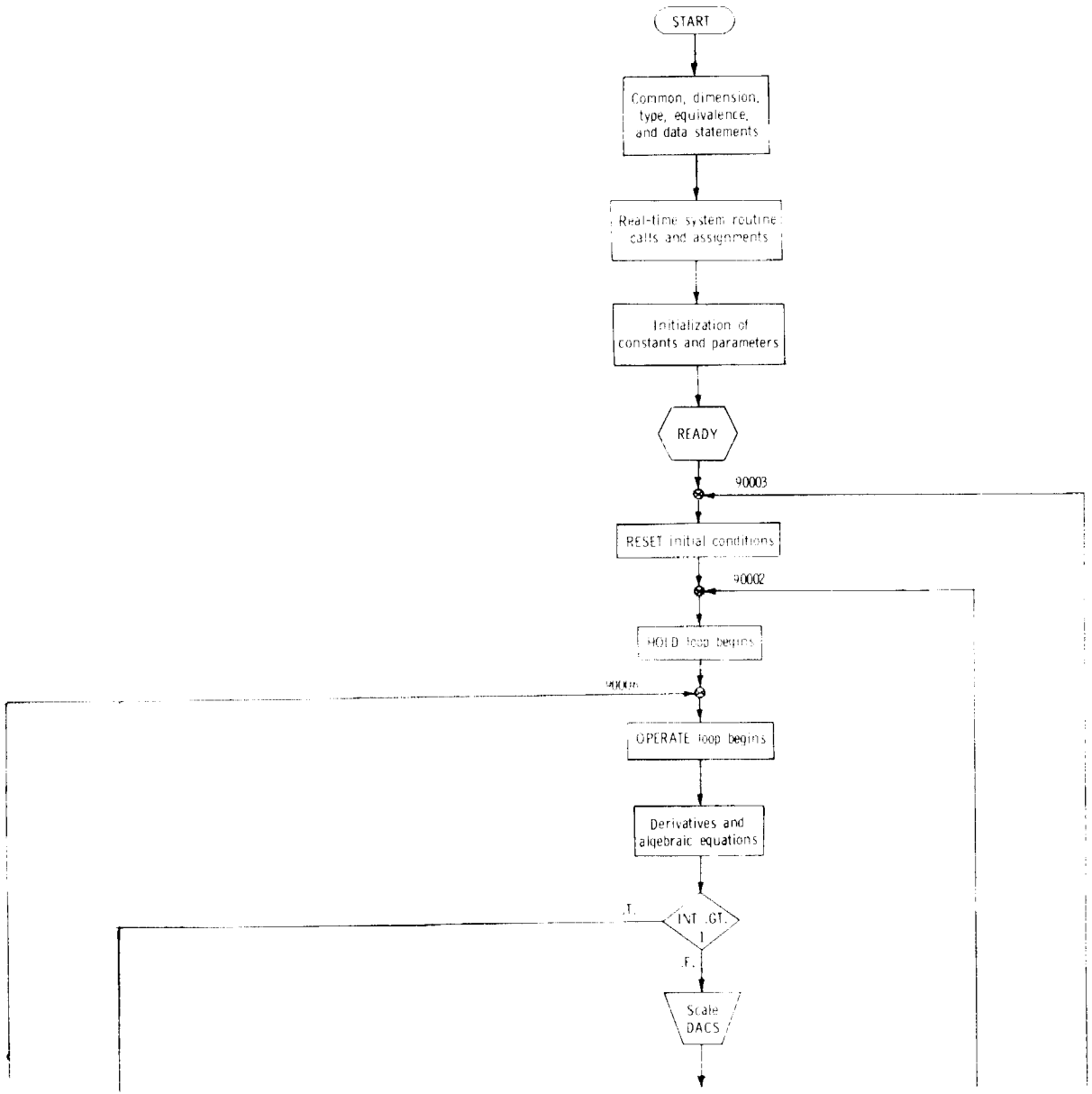


Figure 2.- Block diagram comparing the implementation of AB-2 to that of LL for "norm in."


START
Common, dimension, type, equivalence, and data statements
Real-time system routine calls and assignments
Initialization of constants and parameters
READY
90003
RESET initial conditions
90002
HOLD loop begins
90001
OPERATE loop begins
Derivatives and algebraic equations
.T.
INT .GT. 1
.F.
Scale DACS
Real-time loop begins
Initialize integrations
Body angular rates and their derivatives stored
Time-history recorders scaled

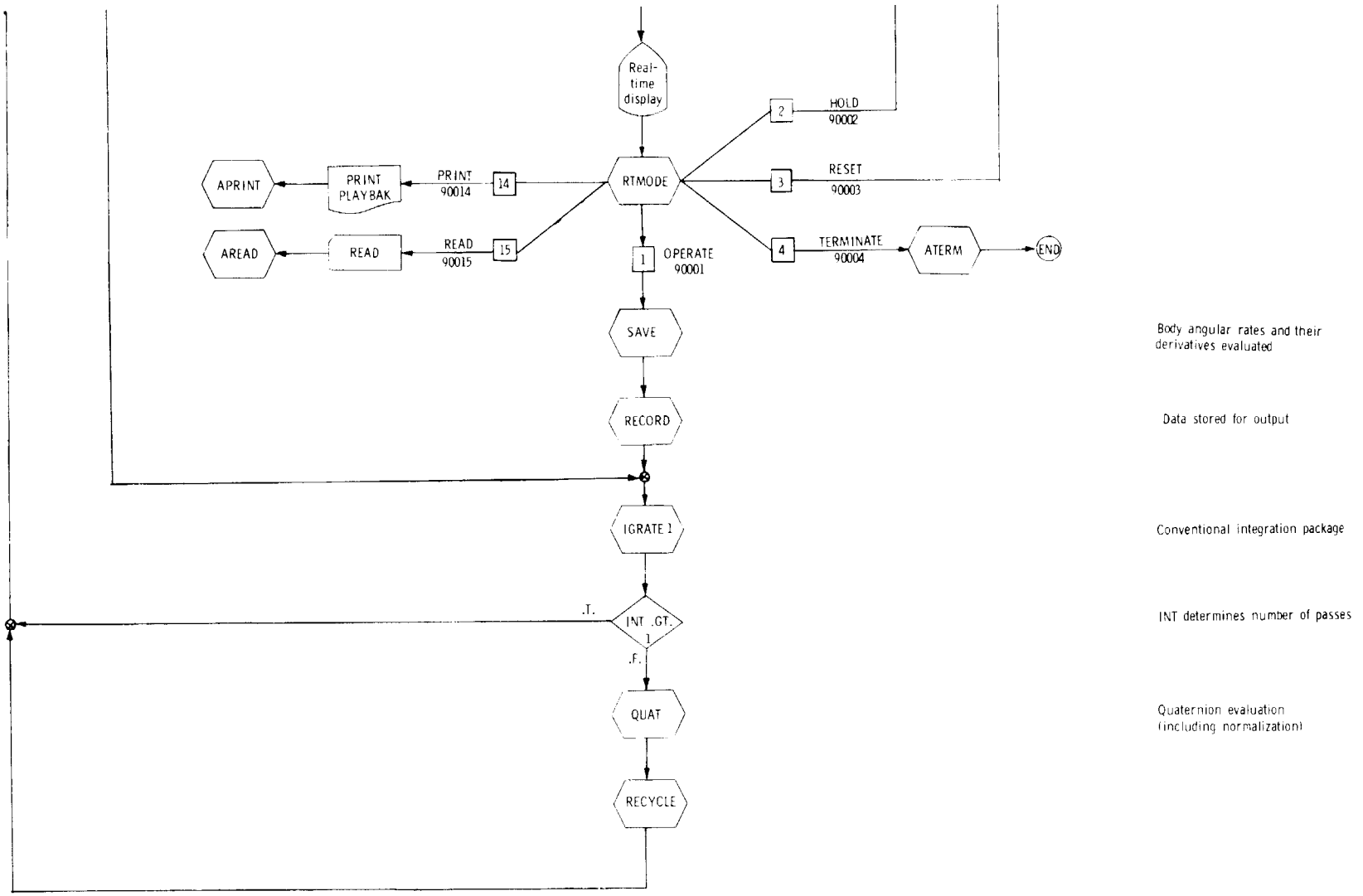
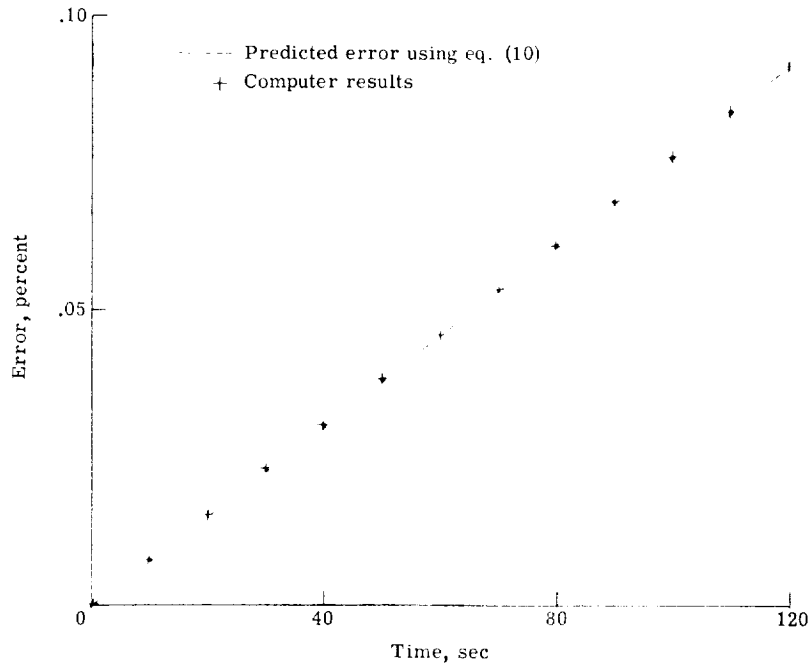
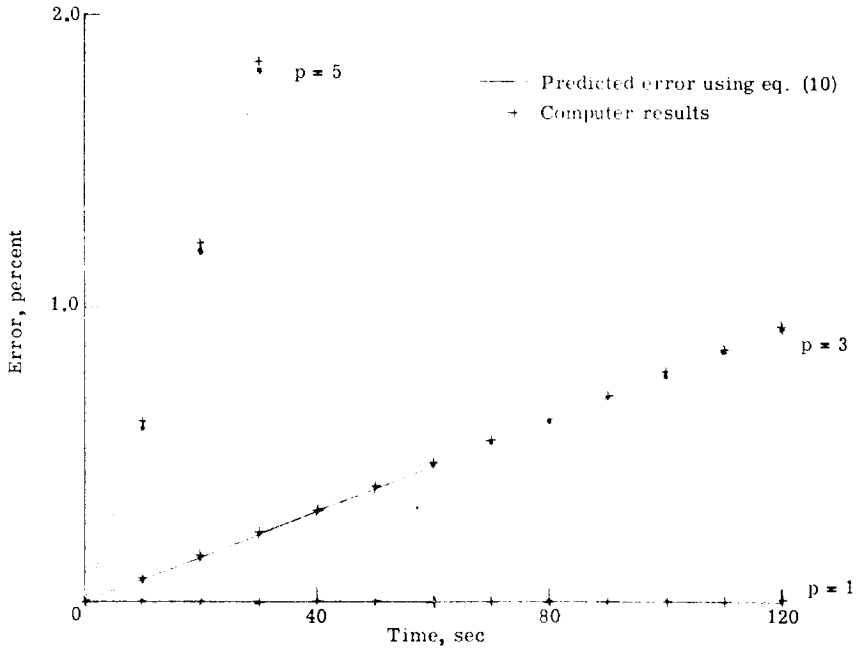


Figure 3.- Flow chart of real-time program employing LL quaternion subroutine package.

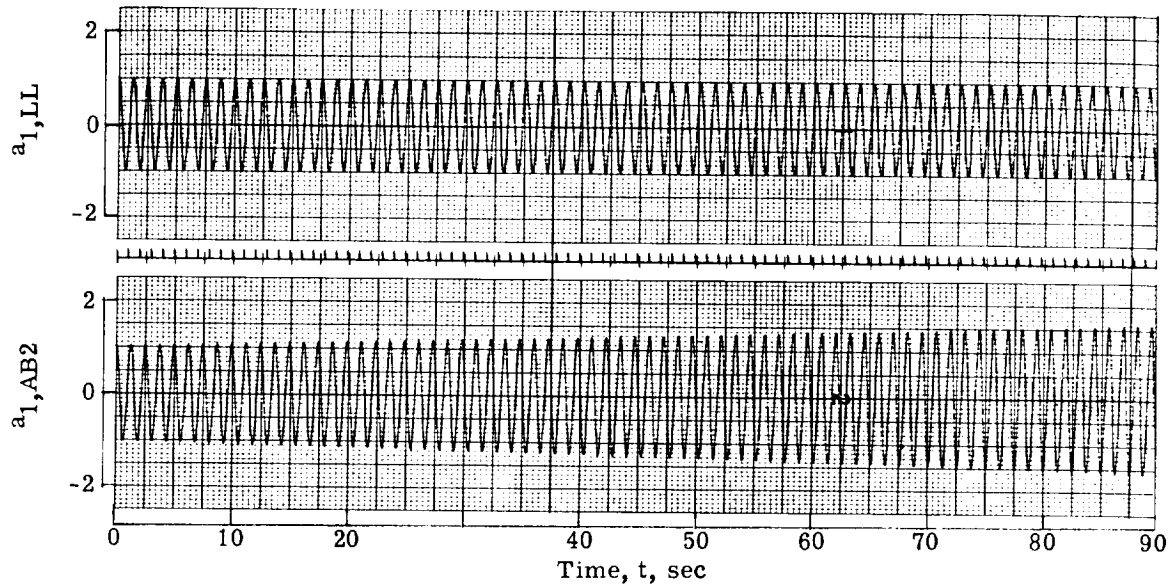


(a) $\mathbf{p} = q = r = 1/\sqrt{3}$ rad/sec and $h = 1/16$ sec.

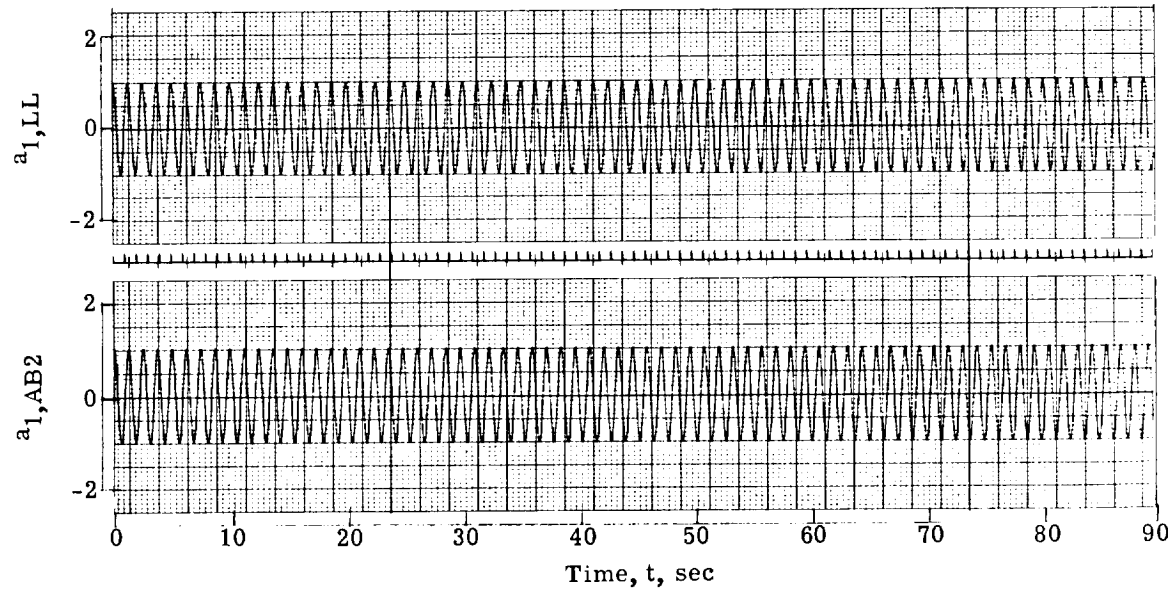


(b) $q = r = 0$ and $p = 1$, 3, and 5 rad/sec; $h = 1/32$ sec.

Figure 4.- Percent error in the norm squared of the Euler parameters using AB-2.



(a) "Norm out."



(b) "Norm in."

Figure 5.- Time-history segment of the quaternion a_1, comparing AB-2 with LL for $h = 1/32$ sec and $p = 10$ rad/sec ($q = r = 0$).

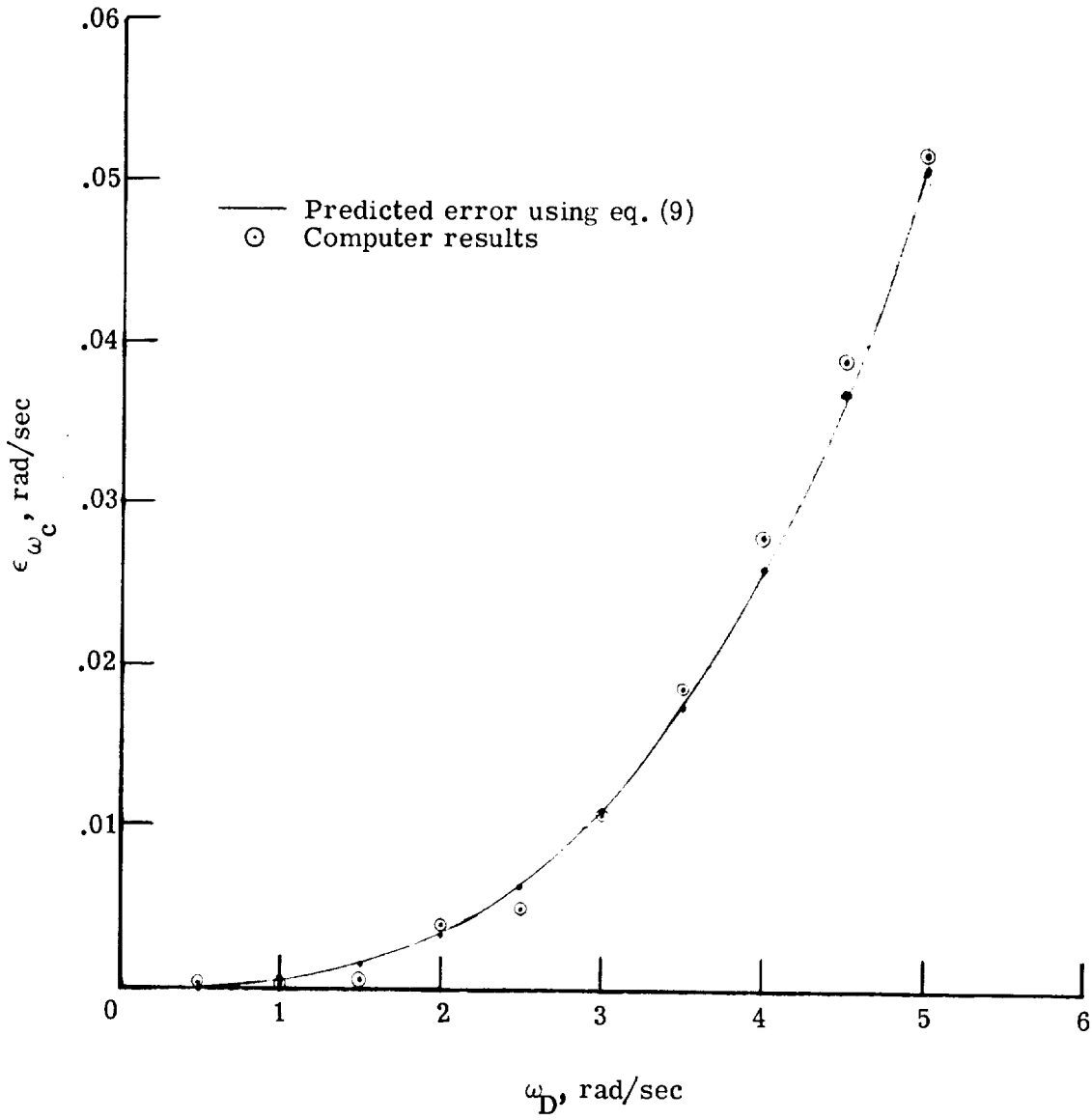


Figure 6.- Error ϵ_{ω_c} in predicted and actual computed frequencies of the quaternions for AB-2 plotted against the correct or desired frequency ω_D for h = 1/32 sec.

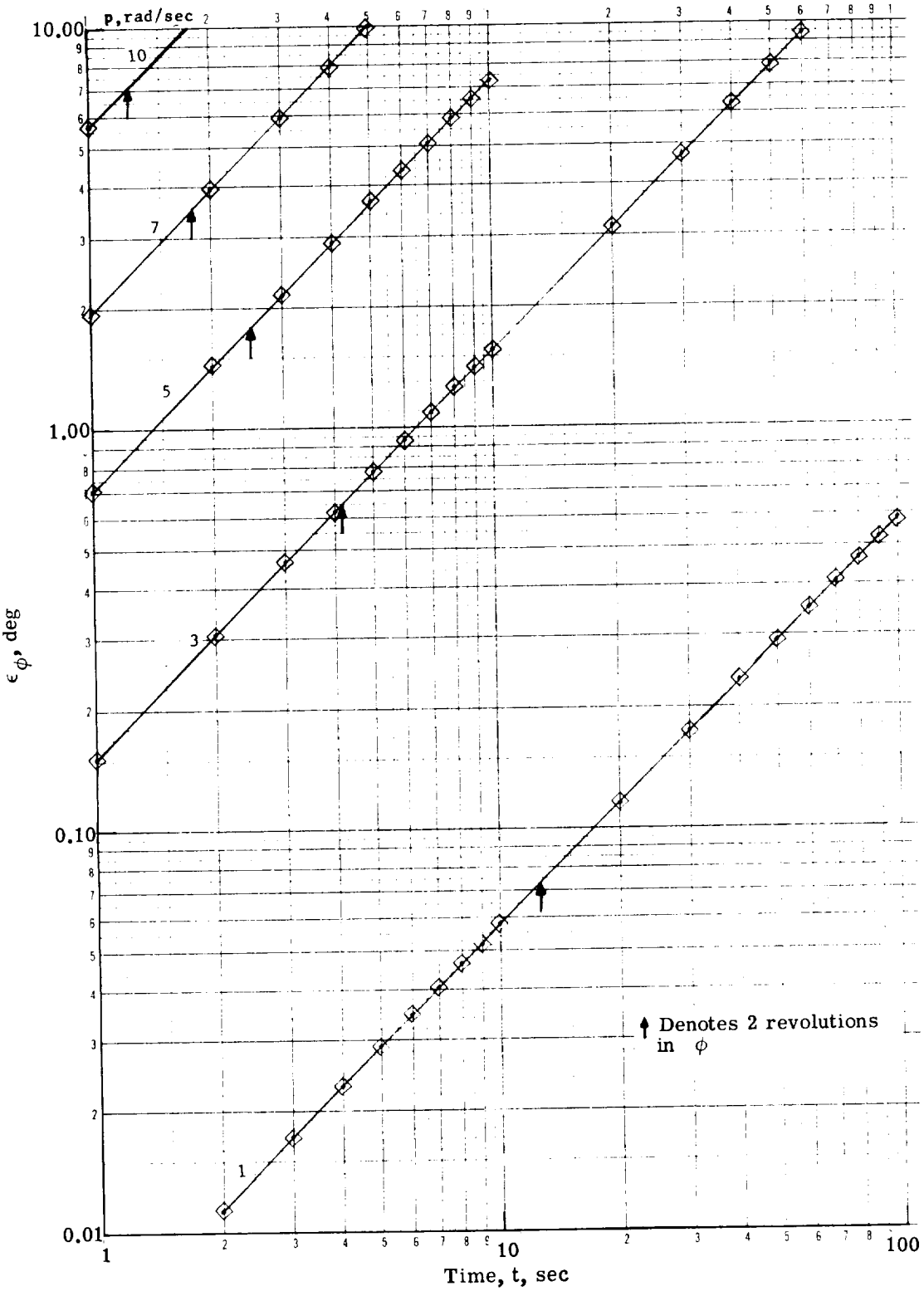


Figure 7.- Roll-angle response error ϵ_ϕ for constant roll rate p (q = r = 0) using AB-2. h = 1/32 sec and "norm in."

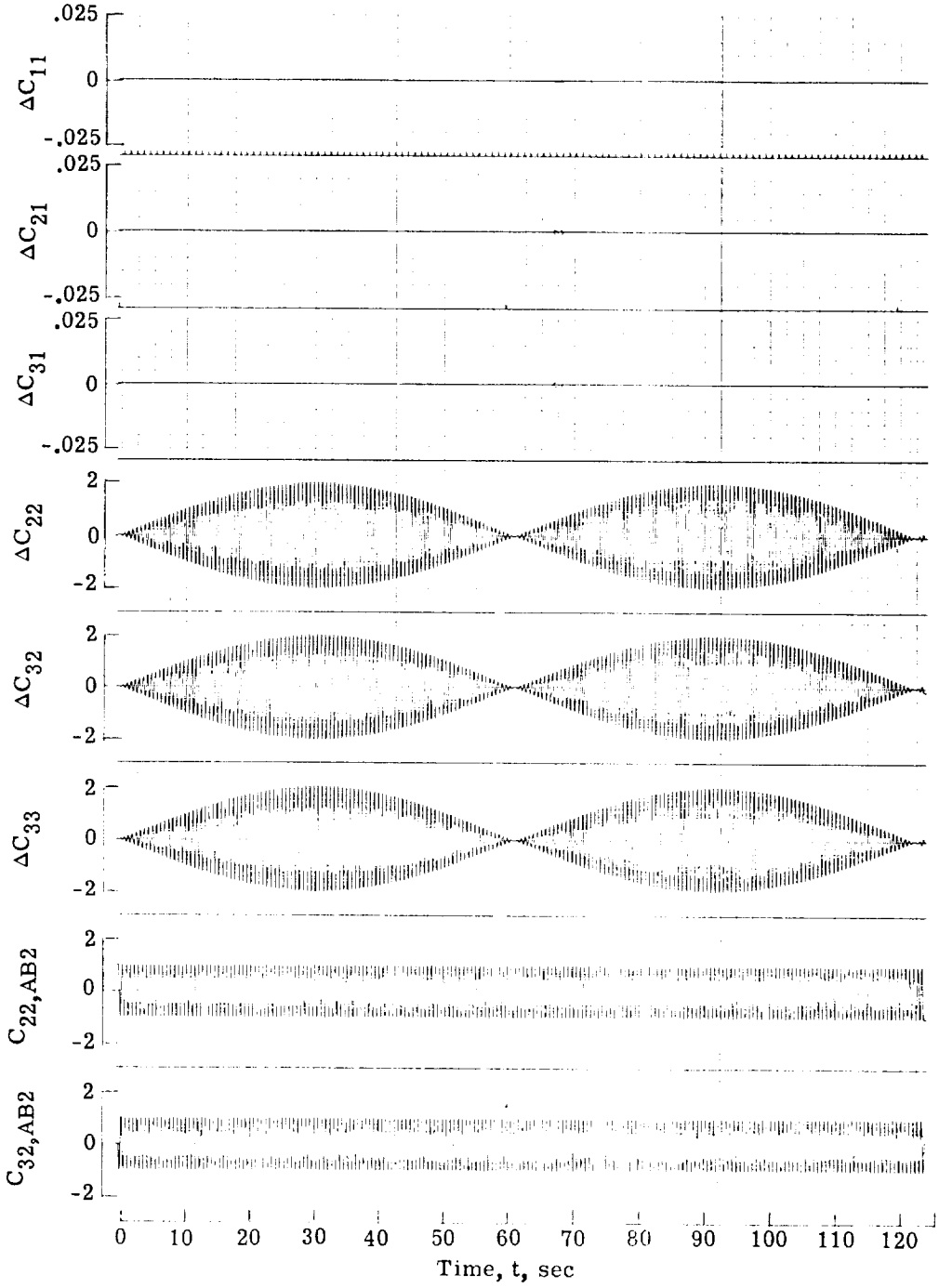
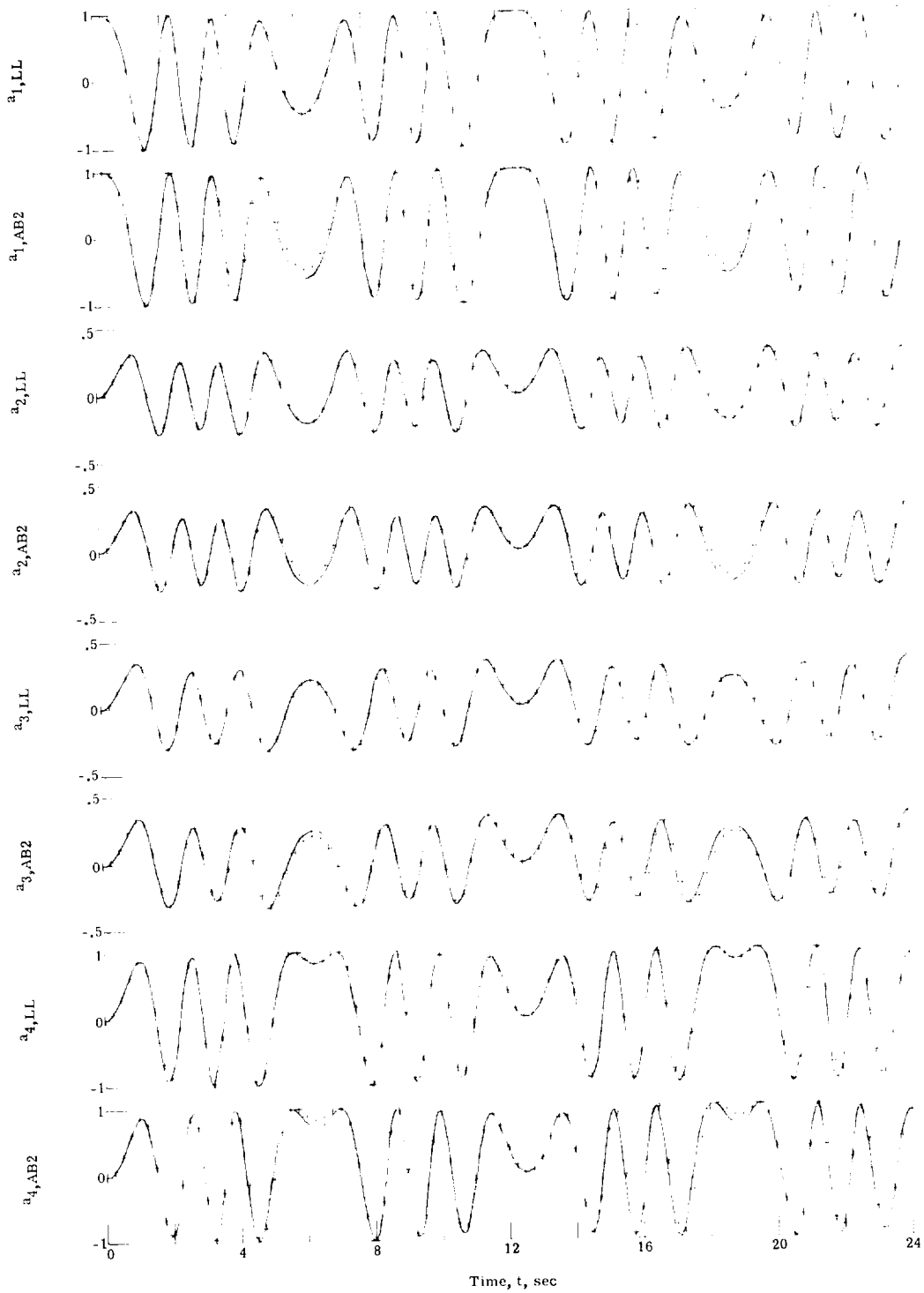
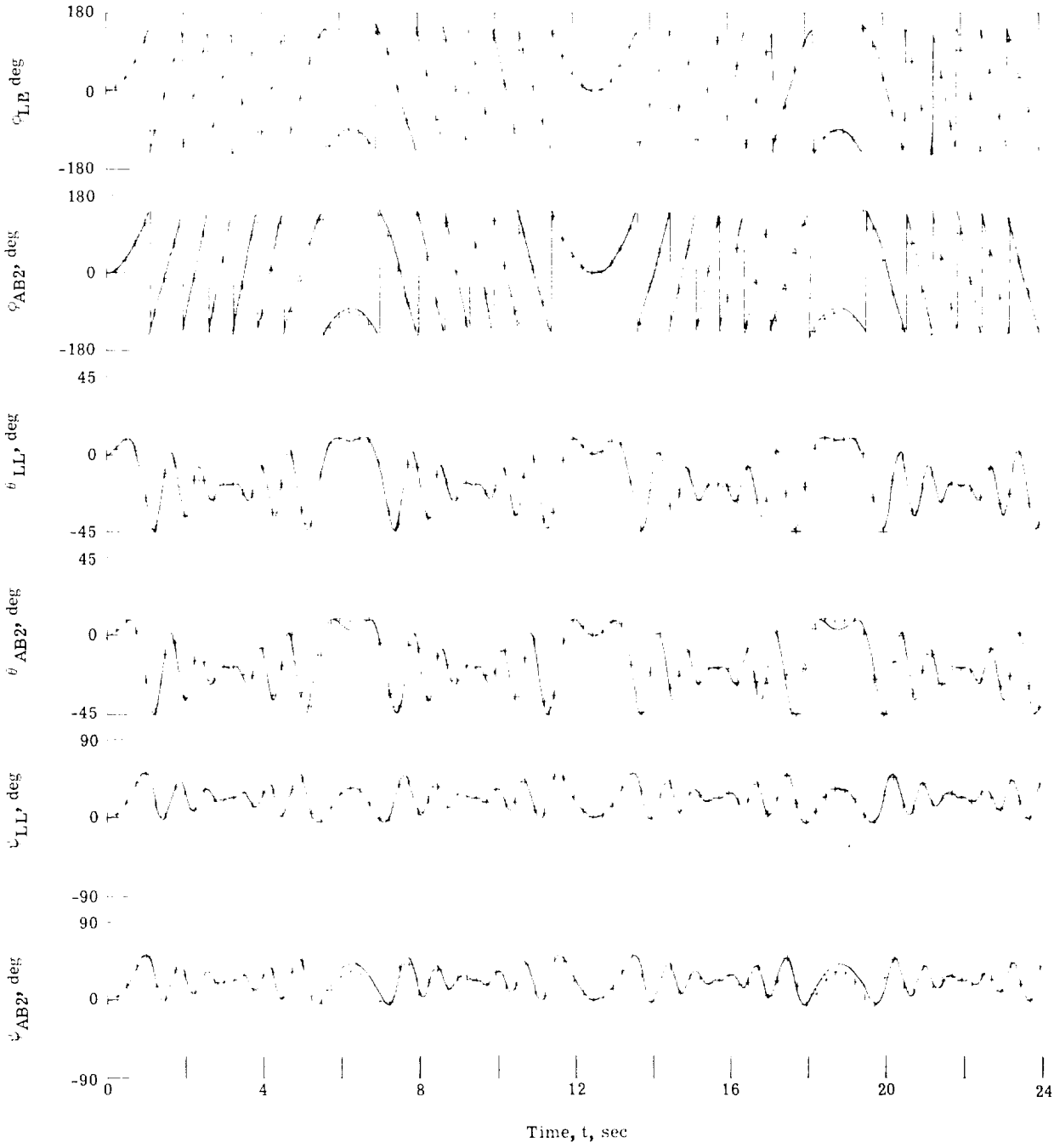


Figure 8.- Differences in solution between AB-2 and LL for direction cosines. h = 1/32 sec; p = 10 rad/sec (q = r = 0); "norm in."



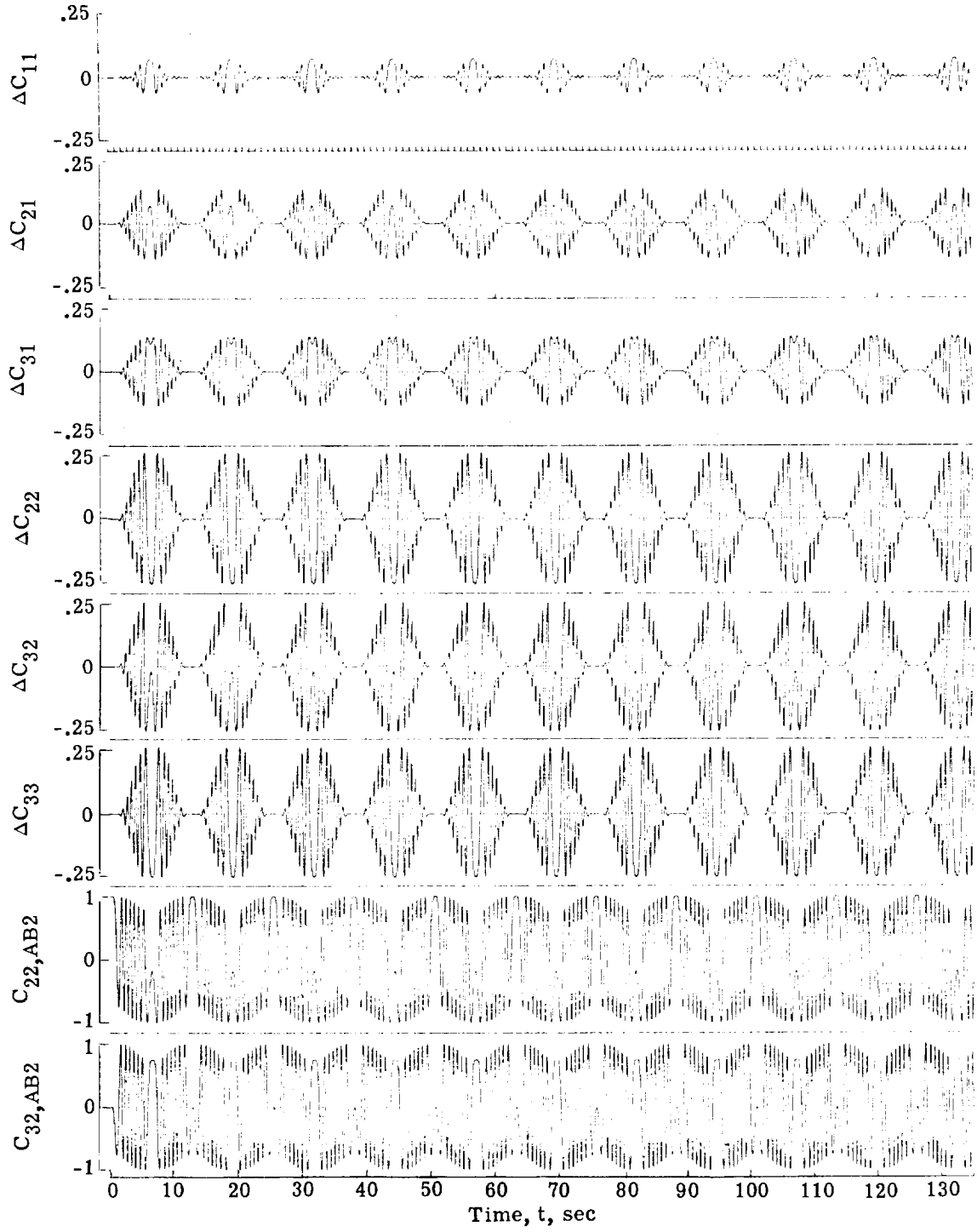
(a) Time histories of the quaternions $a_1, \ldots, a_4$ for AB-2 and LL. (The plus symbol denotes solution of RK-7 used as independent check.)

Figure 9.- Sinusoidal inputs for h = 1/32 sec and "norm in."



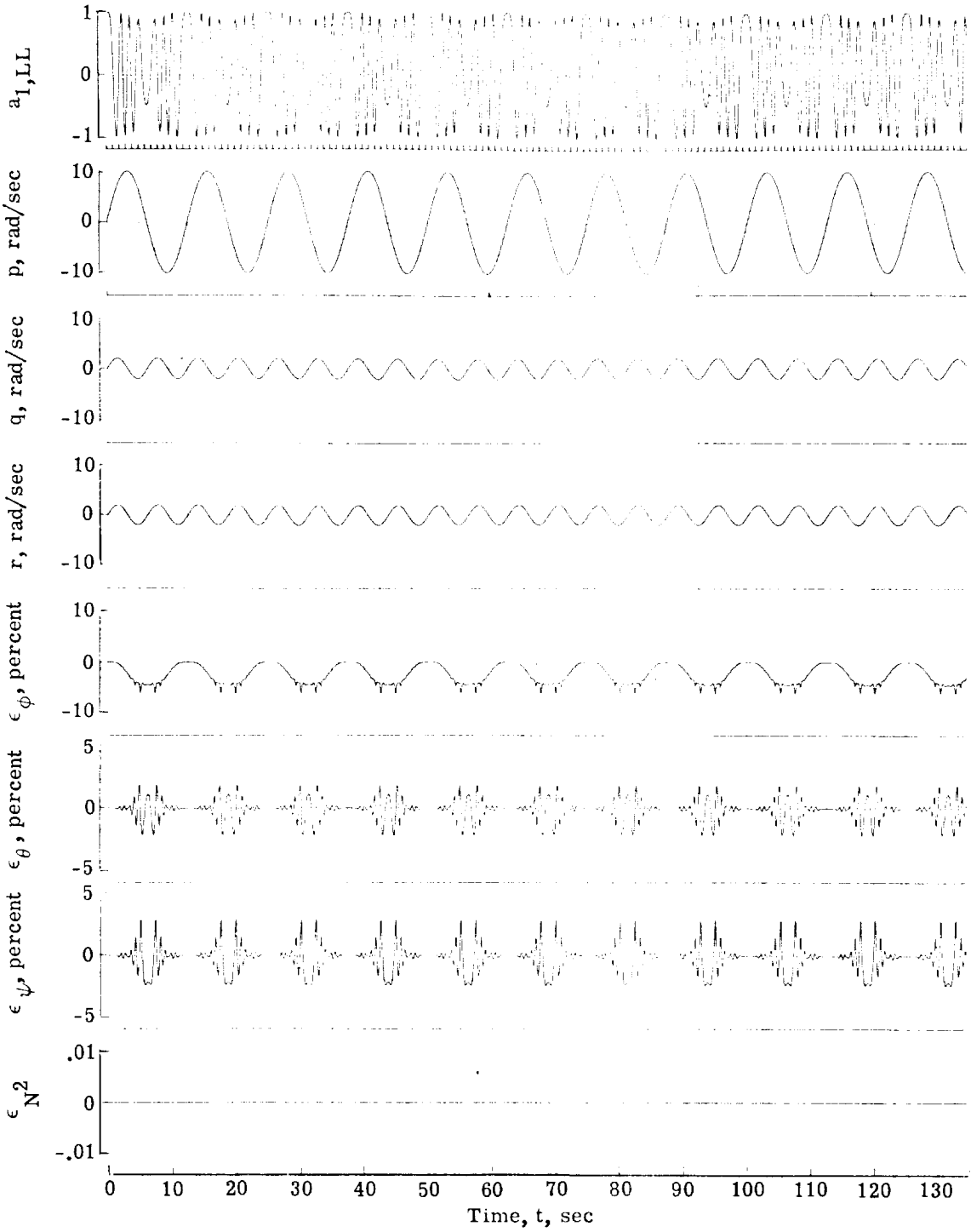
(b) Time histories of the Euler angles ϕ, θ, and ψ for AB-2 and LL. (The plus symbol denotes solution of RK-7 used as independent check.)

Figure 9.- Continued.



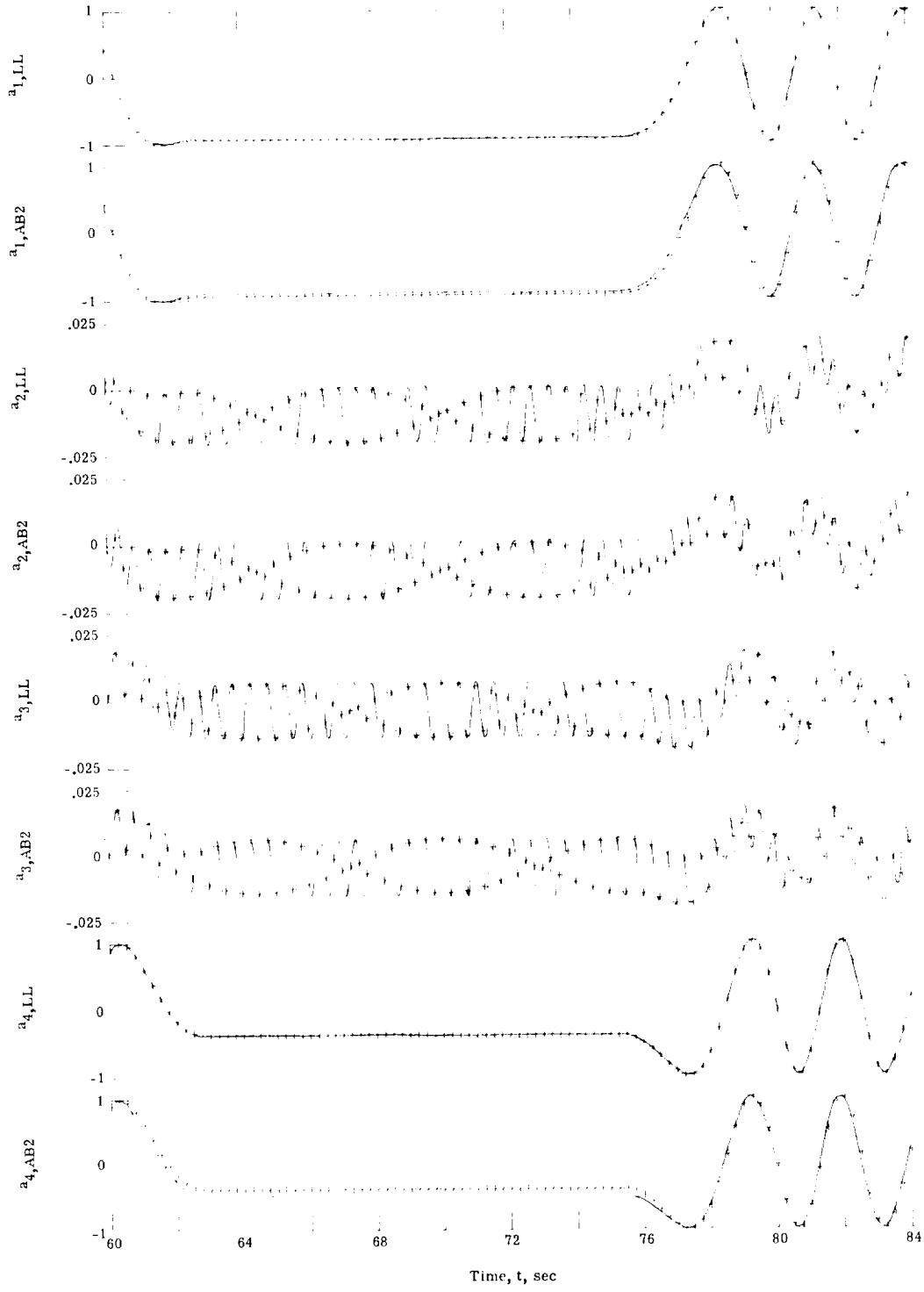
(c) Differences in solution between AB-2 and LL for direction cosines.

Figure 9.- Continued.



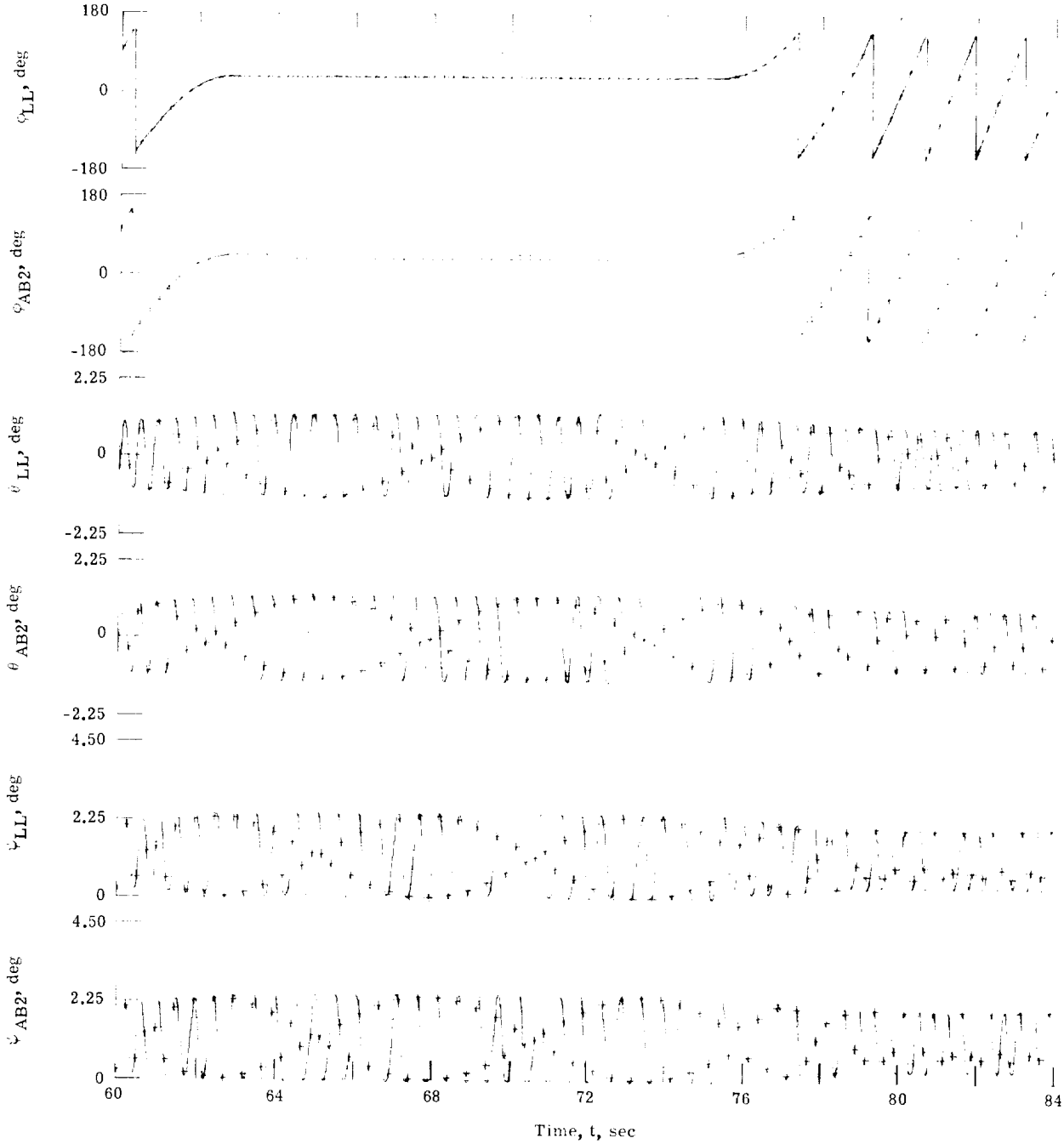
(d) Differences in solution between AB-2 and LL for Euler angles.

Figure 9.- Concluded.



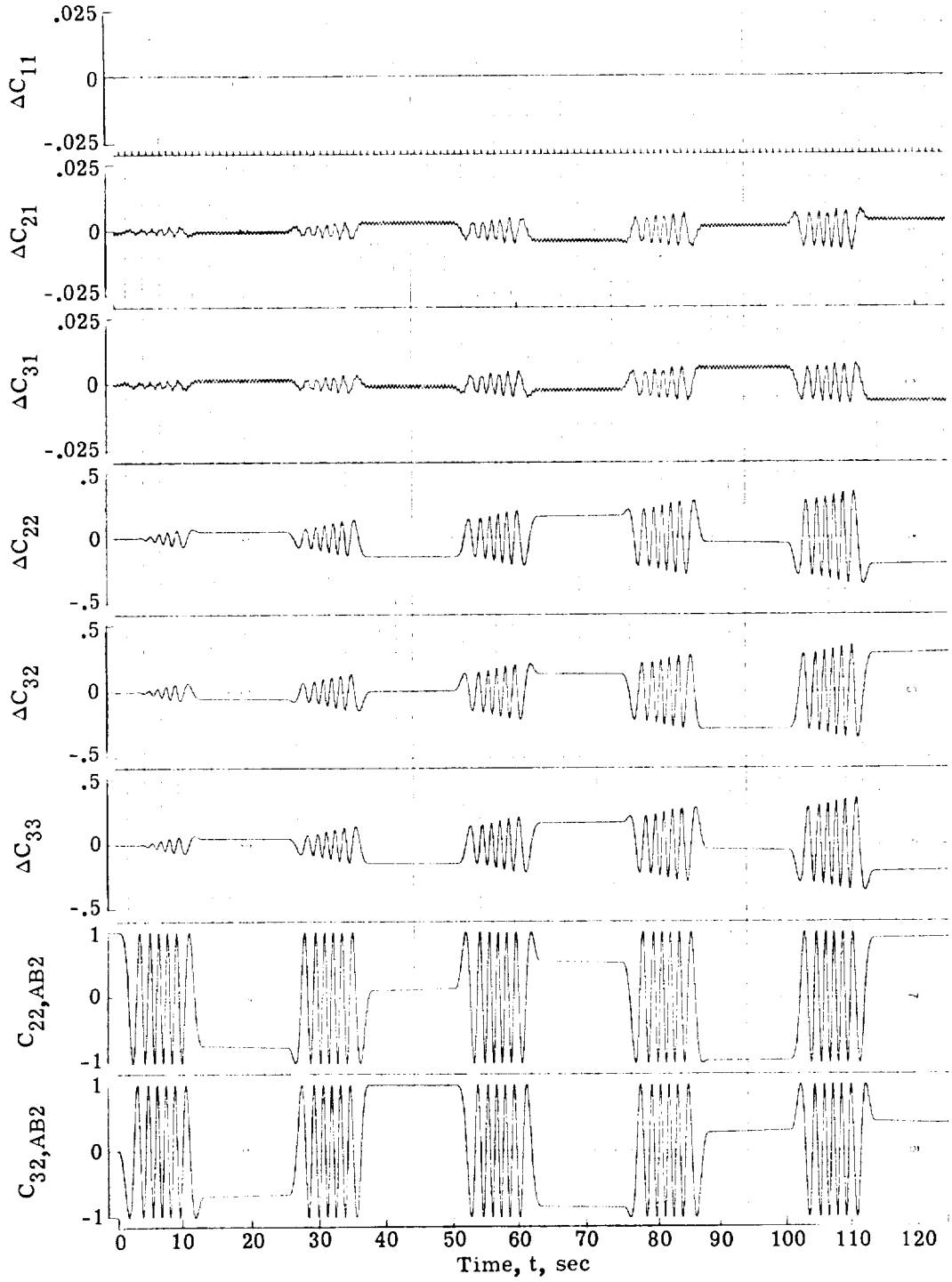
(a) Time histories of the quaternions $a_1, \ldots, a_4$ for AB-2 and LL. (The plus symbol denotes solution of RK-7 used as independent check.)

Figure 10.- Sinusoidal pulse inputs for h = 1/32 sec and "norm in."



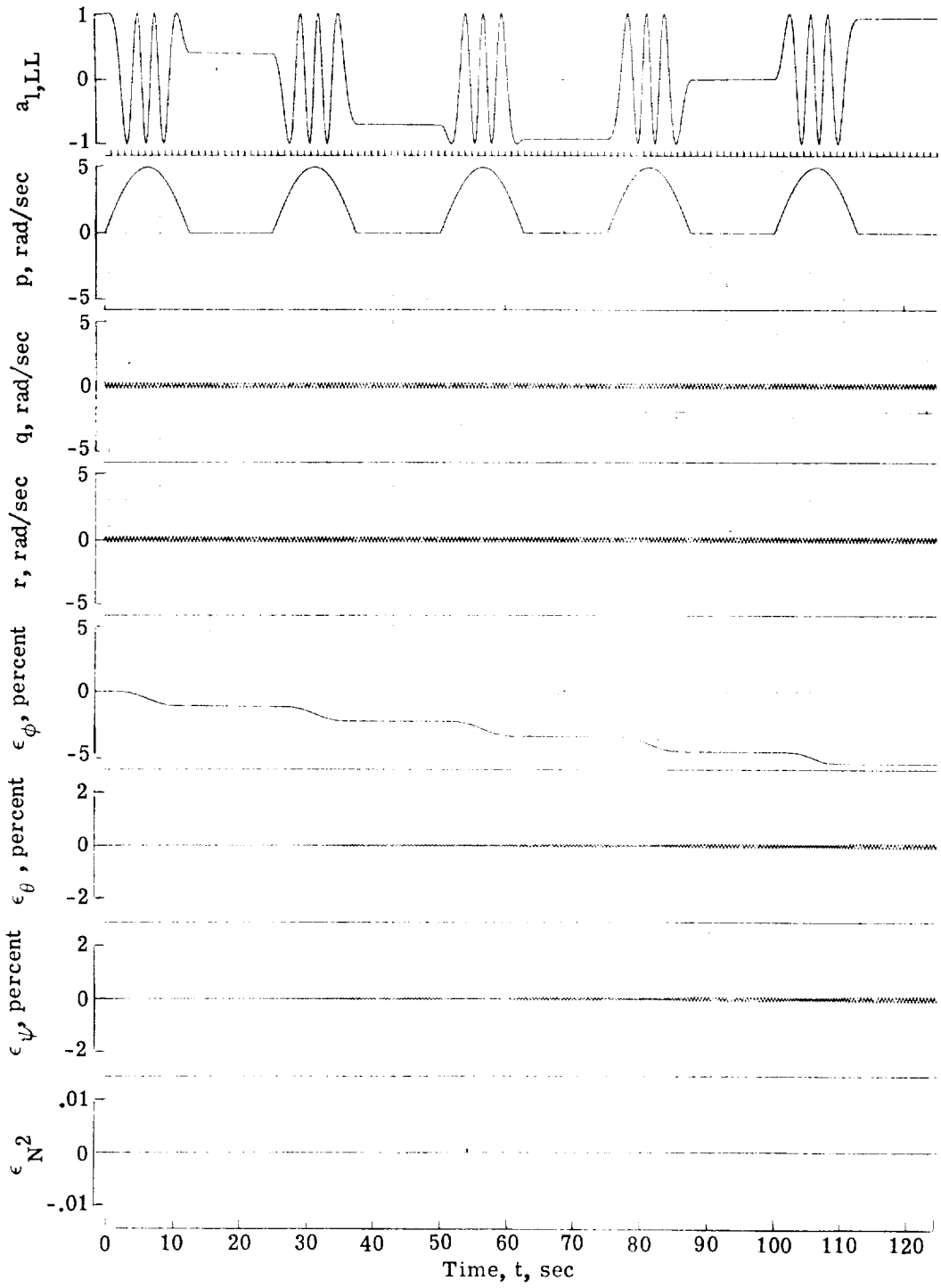
(b) Time histories of the Euler angles ϕ, θ, and ψ for AB-2 and LL. (The plus symbol denotes solution of RK-7 used as independent check.)

Figure 10.- Continued.



(c) Differences in solution between AB-2 and LL for direction cosines.

Figure 10.- Continued.



(d) Differences in solution between AB-2 and LL for Euler angles.

Figure 10.- Concluded.

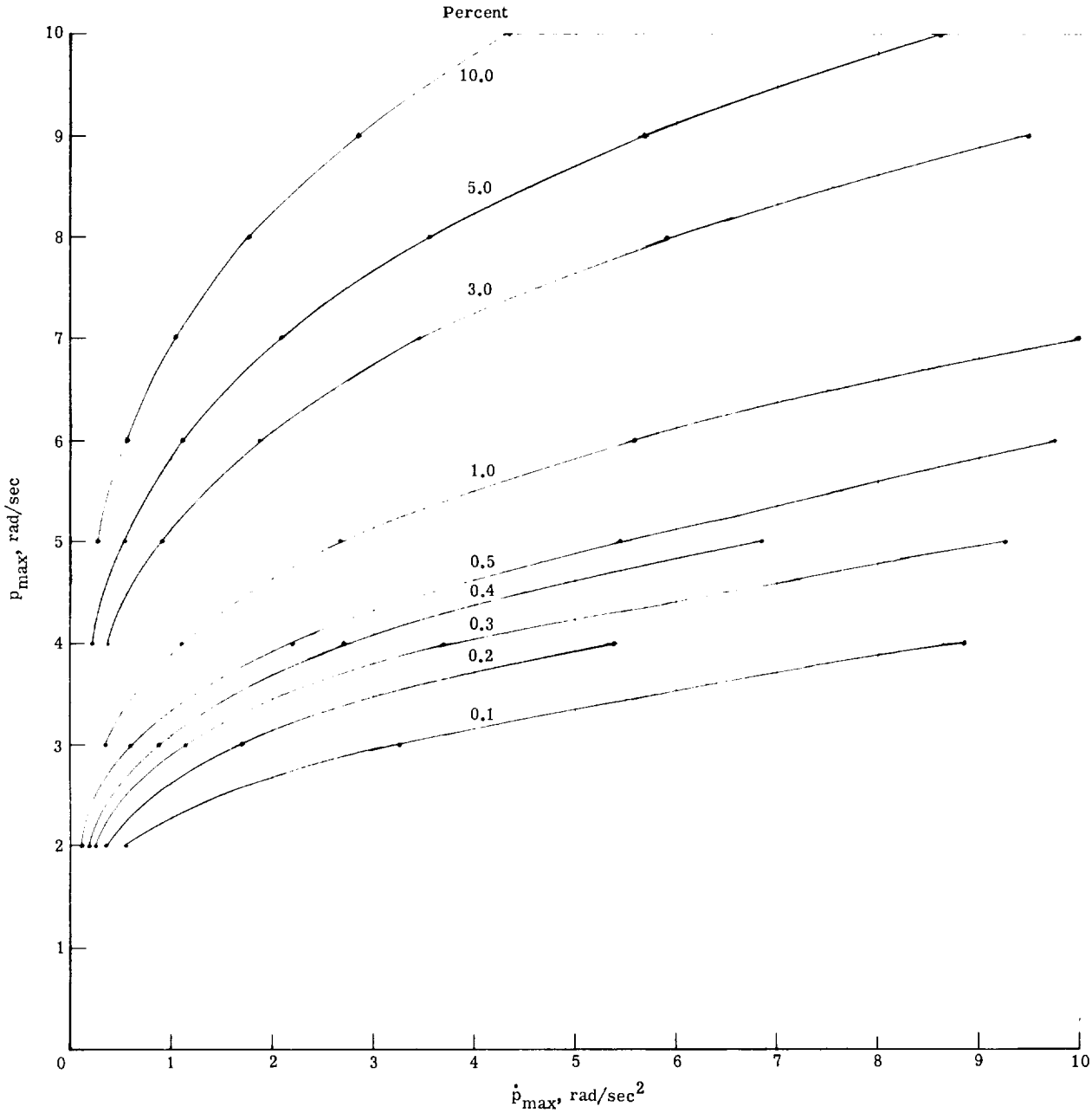


Figure 11.- Curves of constant error in roll angle ϕ for two consecutive sinusoidal pulses in roll rate p using AB-2. h = 1/32 sec and "norm in."

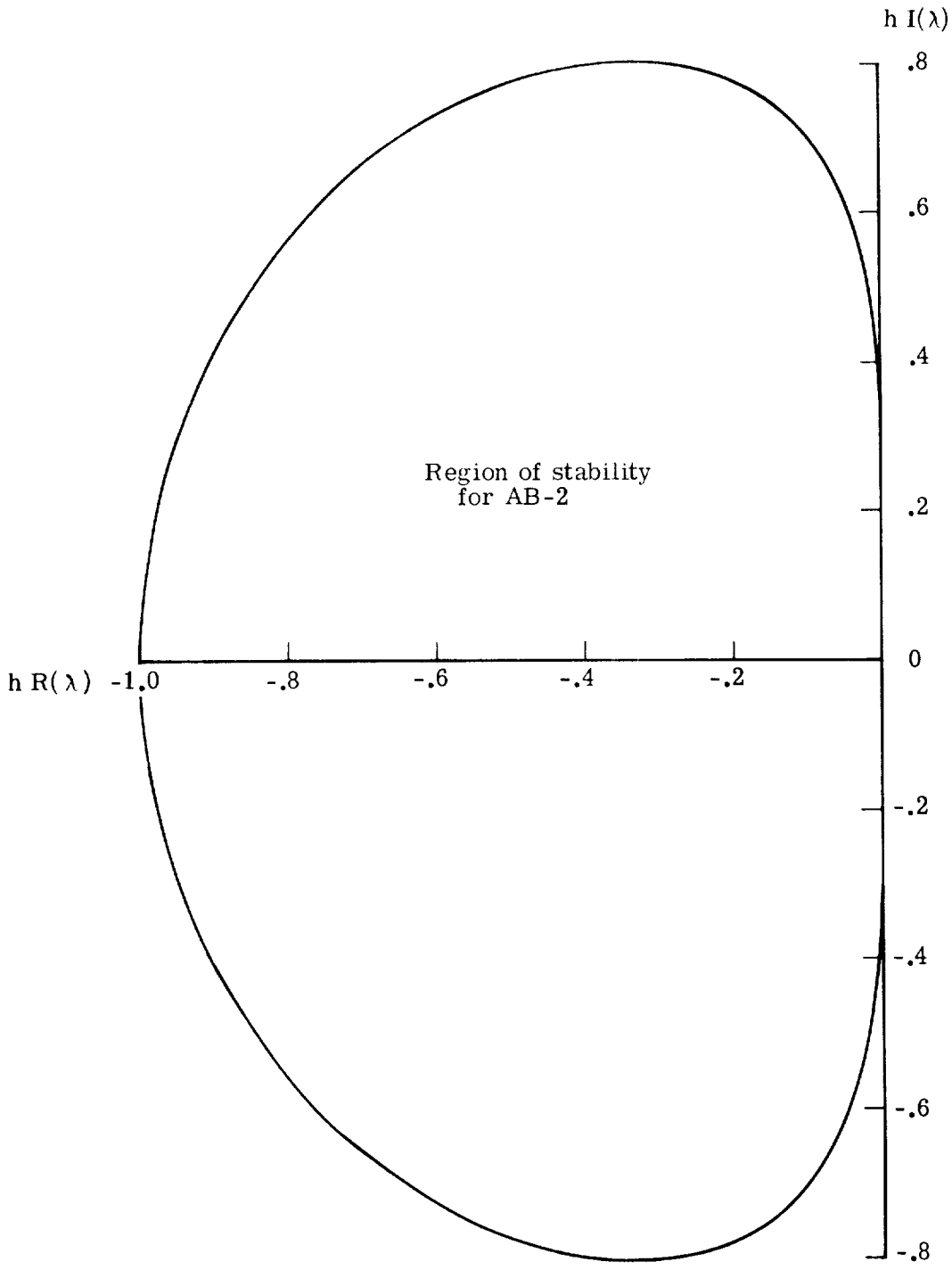
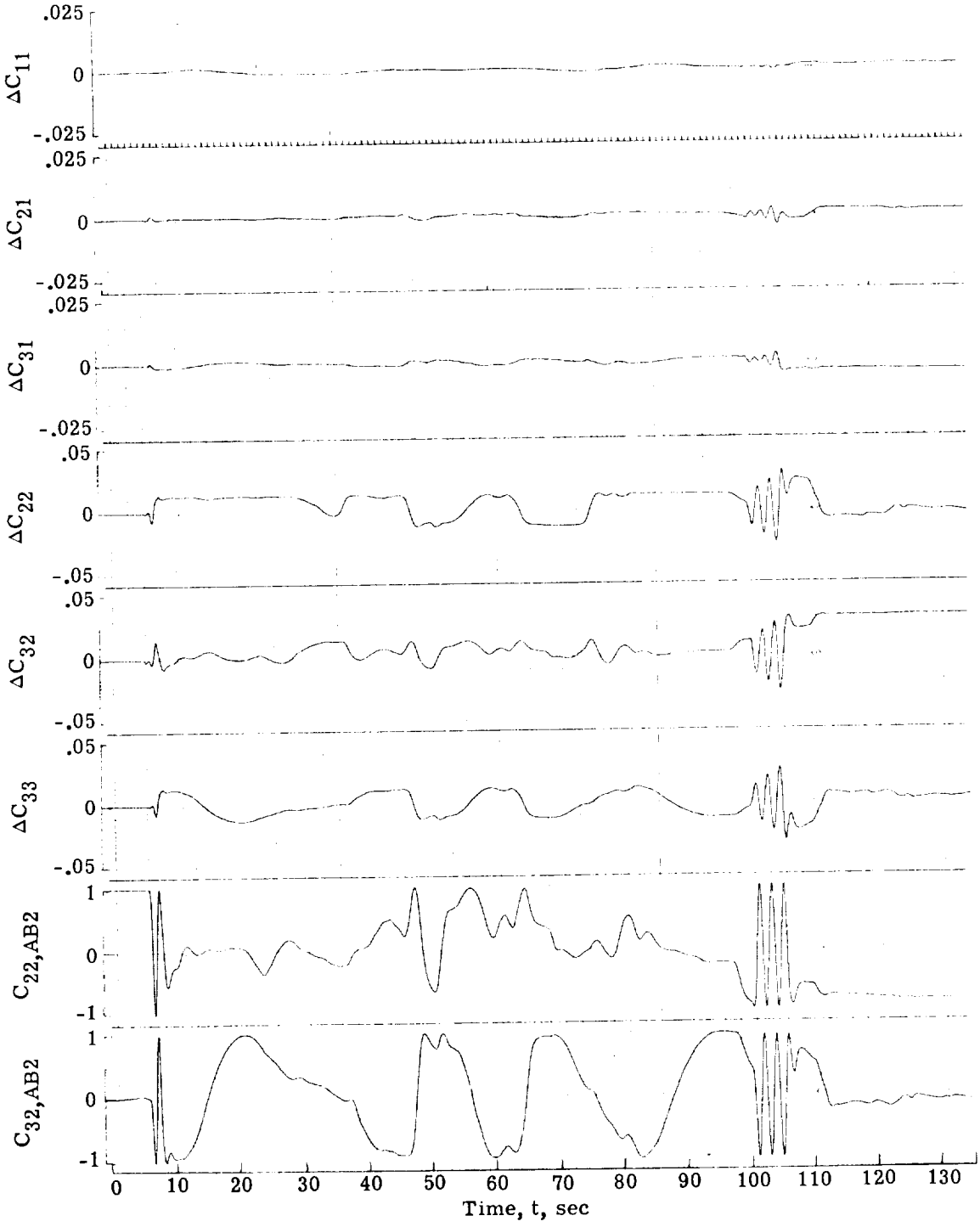
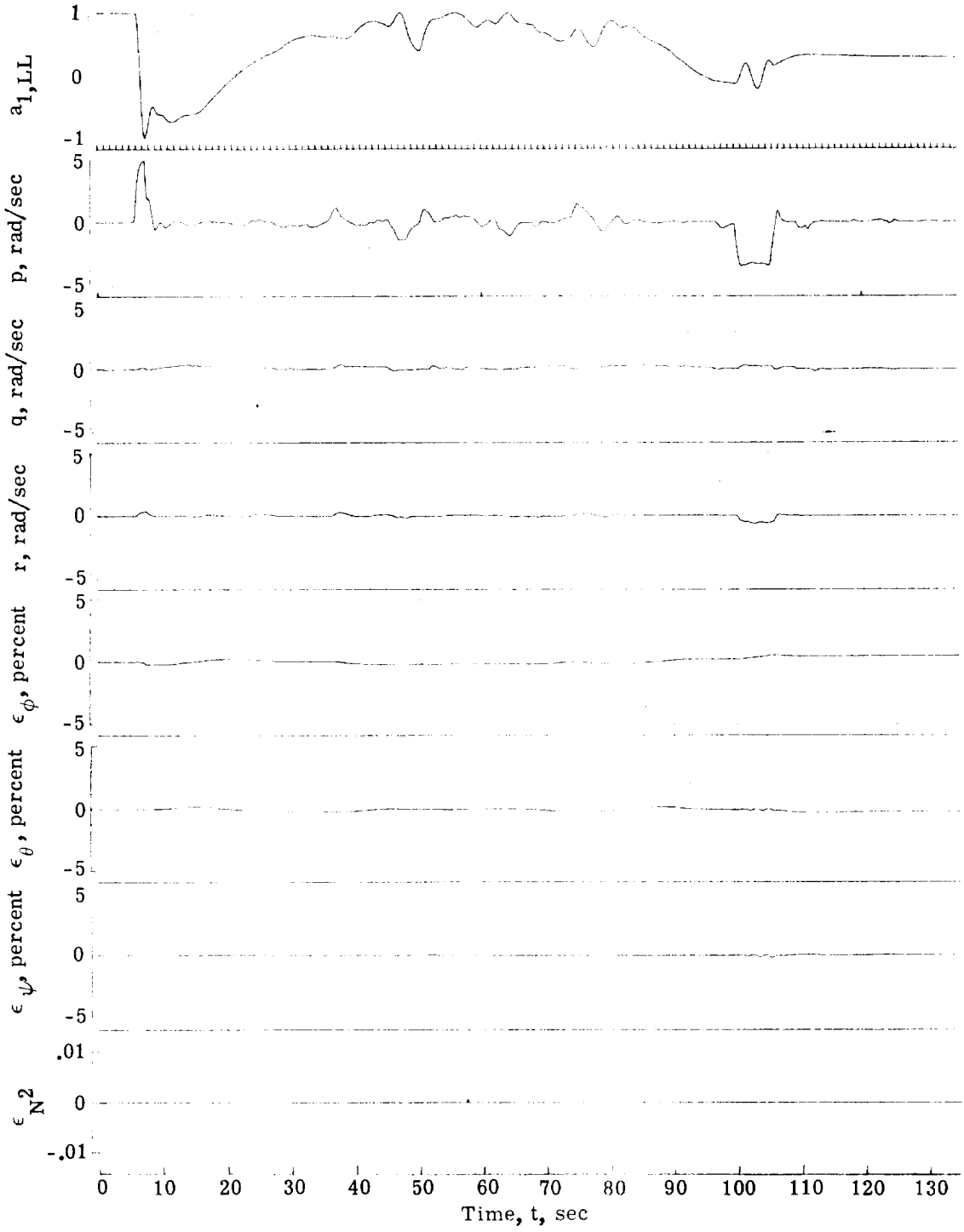


Figure 12.- Stability boundary for AB-2 method.



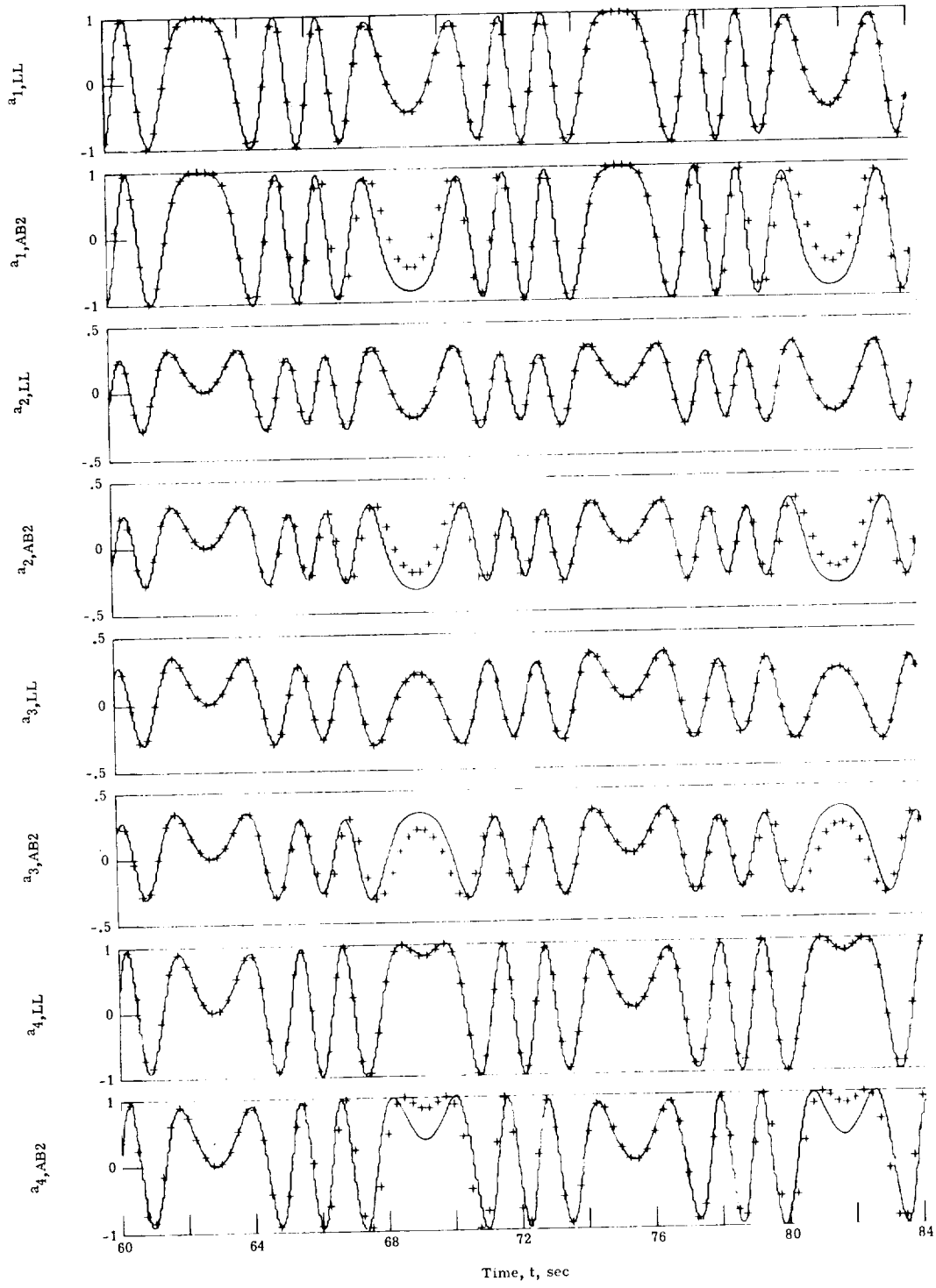
(a) Direction cosines.

Figure 13.- Differences in solution between AB-2 and LL for taped piloted run. h = 1/32 sec and "norm in."



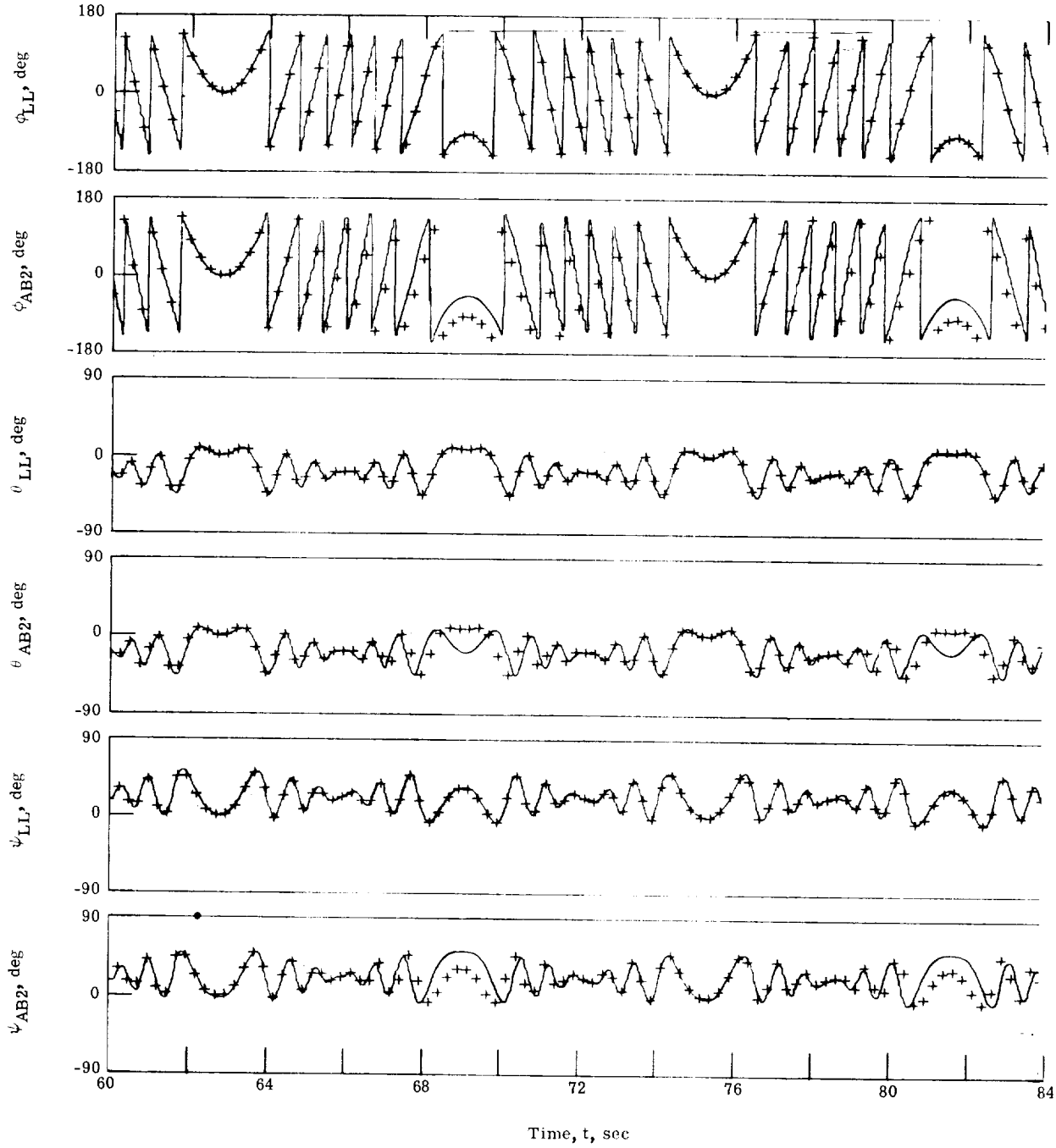
(b) Euler angles.

Figure 13.- Concluded.



(a) Time histories of the quaternions $a_1, \ldots, a_4$ for AB-2 and LL. (The plus symbol denotes solution of RK-7 used as independent check.)

Figure 14.- Sinusoidal inputs for h = 1/16 sec and "norm in."



(b) Time histories of the Euler angles ϕ, θ, and ψ for AB-2 and LL. (The plus symbol denotes solution of RK-7 used as independent check.)

Figure 14.- Concluded.

NASA-Langley, 1973 — 19 L-8954