NASA TECHNICAL REPORT

NASA TR R-304

NASA TR R-304
C. 1

DIRECTION COSINE COMPUTATIONAL ERROR

by John W. Jordan
Electronics Research Center
Cambridge, Mass.

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • MARCH 1969

DIRECTION COSINE COMPUTATIONAL ERROR

By John W. Jordan

Electronics Research Center
Cambridge, Mass.

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – CFSTI price $3.00

DIRECTION COSINE COMPUTATIONAL ERROR

By John W. Jordan
Electronics Research Center

SUMMARY

Strapdown inertial systems possess a potential advantage over more conventional inertial systems since they do not require a mechanical gimbal structure to maintain a stable coordinate reference. Instead, strapdown systems replace the gimbal structure by a direction cosine matrix which is contained in and updated by the system computer. To assess the performance of a strapdown system or to determine the computer requirements, a detailed study of the computational error introduced by the computer is necessary. This report is concerned with the techniques which may be employed to estimate the computational error and its effects on system performance. Although focused upon a particular problem of practical importance, many of the techniques have a wider applicability to aerospace software in general. The report is divided into the following sections:

1. Introduction - The computer must solve the matrix differential equation $\dot{B} = \Omega B$ where B is the computed direction cosine matrix and Ω is a skew-symmetric angular rate matrix, the elements of which are determined by the system gyroscopes. An error matrix (z) is defined which provides a link between the direction cosine error and system performance. The computer-generated error is divided into two categories: (a) Algorithm Error, which is due to the approximate nature of the numerical formulas used, and (b) Wordlength Error, which is due to a finite computer wordlength.

2. Algorithm Error - A differential equation is derived for the propagation of the error (z) matrix. It applies to a general digital computational process with an arbitrary rate input. Since B should be an orthogonal matrix, it is possible to include a correction routine in the airborne computer program so that B will remain orthogonal despite computational error. A second differential equation is derived for the propagation of the z matrix when an orthogonality correction is used.

   Closed form analytical solutions are obtained for both error differential equations, provided that the angular rate matrix is self-commutative.

   A computable solution which allows the determination of the z matrix for any arbitrary angular rate input is developed. All solutions are verified by simulation.

A compensation technique is devised (and verified by simulation) which greatly enhances the performance of low order algorithms. In many cases, this technique will reduce both the error and computer requirements.

3. Wordlength Error - Wordlength error is treated from the statistical viewpoint of Henrici. The theory is developed for linear matrix differential equations. The direction cosine problem is then used as a specific example. The results of Henrici are extended to include different computer number systems (i.e., sign-magnitude or complement representations) and different organizations of the computer's arithmetic unit. The emphasis is upon a computable solution which may be applied to problems for which analytical solutions are either not possible or are not practical. The solution includes both fixed- and floating-point machines. The computable solution is verified by simulation.

4. System Performance - The computable solutions provide the algorithm and wordlength error matrices for any arbitrary vehicle rate profile. It is possible to incorporate these solutions into a conventional error analysis program in order to determine overall system performance. The computational error is handled in a manner completely analogous to the inertial instrument error and system performance is expressed by identical criteria.

A method (susceptible to automation) is presented for the "optimum" design of aerospace software. Although the approach is again general in nature, it is related to the problem of selecting the "best" direction cosine algorithm for a particular mission.

I. INTRODUCTION

The Apollo vehicle that lands on the moon will have as a backup system a "strapdown" inertial navigator. The term "strapdown" indicates that the accelerometers of the inertial system are rigidly connected ("strapped down") to the parent vehicle. The acceleration that they measure will be in body coordinates, i.e., dependent upon the instantaneous orientation of the vehicle, and not in the coordinate frame in which the navigation equations are solved. In order to remove the effect of vehicle rotations, a coordinate transformation matrix B(t) is calculated by the computer so that the measured acceleration vector in body coordinates a_B may be resolved into the navigational frame as a_N:

$$\underline{a}_N(t) = B^T(t)\underline{a}_B(t) \qquad (1)$$

The transposed matrix is used for convenience in the sequel. The matrix B(t) is 3×3 and contains nine elements called the direction cosines. Their time rate of change is given by

$$\dot{B}(t) = \Omega(t)B(t) \tag{2}$$

where $\Omega(t)$ is a skew-symmetric matrix of body angular rates as measured by the system gyroscopes

$$\Omega(t) = \begin{bmatrix} 0 & \omega_Z(t) & -\omega_Y(t) \\ -\omega_Z(t) & 0 & \omega_X(t) \\ \omega_Y(t) & -\omega_X(t) & 0 \end{bmatrix} . \tag{3}$$

The computer solves the matrix differential equation (Eq. (2)) in real time. This computation must be done many times a second. To evaluate system performance, it is necessary to estimate the error introduced by the computational process. This report is not intended as a comparative analysis of the many procedures which have been proposed for solving the differential equations. Rather, it is an attempt to formulate a consistent analysis technique which may be applied to the different algorithms and missions.

Error Criteria

The direction cosines of a strapdown inertial system are used to solve the equation

$$\underline{a}_N(t) = B^T(t)\underline{a}_B(t) \tag{4}$$

where

$\underline{a}_B$ = the vector acceleration of the vehicle in body coordinates,

$\underline{a}_N$ = the vector acceleration of the vehicle in navigational coordinates,

B = the direction cosine matrix which represents the transformation from body to navigational coordinates.

At this point, the manner in which the computer solves the continuous Eq. (4) is not of concern. A digital computer will use a discrete formulation and this will introduce some error. To separate the effect of different error sources, it will be assumed that the computer is capable of solving Eq. (4) exactly, but that the direction cosines are not accurately known. In other words, the computer solves (exactly):

$$\underline{\tilde{a}}_N = \hat{B}^T(t)\underline{a}_B \tag{5}$$

where $\hat{B}(t)$ represents a "best estimate" of the true direction cosine matrix $B(t)$, and $\underline{\tilde{a}}_N$ is the measured (not true) acceleration in the navigational frame. Defining

$$E(t) = \hat{B}(t) - B(t), \tag{6}$$

the acceleration error is given by

$$\delta\underline{a}_N = E^T\underline{a}_B \ . \tag{7}$$

Since

$$\underline{a}_B = B\underline{a}_N \ ,$$

then

$$\delta\underline{a}_N = E^T B\underline{a}_N \ .$$

Defining

$$Z = B^T E$$

the two basic error equations are:

$$\delta\underline{a}_N = E^T\underline{a}_B \tag{8}$$

$$\delta\underline{a}_N = Z^T\underline{a}_N \ . \tag{9}$$

These equations provide two alternate descriptions for the effect of the direction cosine error. They are illustrated in Figure 1.

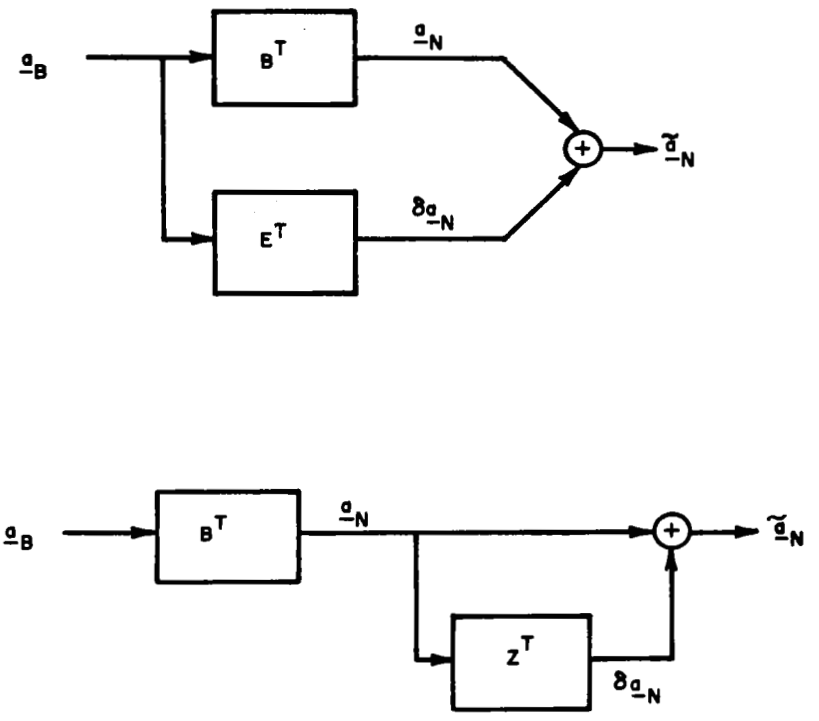


Figure 1.- Effect of direction cosine error

Integration of the acceleration error described by either Eq. (8) or (9) for a specific mission profile will determine the errors in position and velocity which results from the direction cosine error. Although this is a fundamental technique for relating the direction cosine errors to system performance, it sometimes suffices to consider the cosine error matrices alone if one is not concerned with overall system analysis but simply the comparative performance of different cosine algorithms.

As a matter of convenience, the Z matrix rather than the E matrix will be considered. Since any matrix is equal to the sum of a symmetric and a skew-symmetric matrix,

$$Z = Z_S + Z_{SS} \tag{10}$$

with the symmetric part equal to

$$Z_S = \frac{1}{2}\left(Z + Z^T\right) \tag{11}$$

and the skew-symmetric part equal to

$$Z_{SS} = \frac{1}{2}\left(Z - Z^T\right). \tag{12}$$

The Z_{SS} matrix will contain three independent elements which will be denoted by

$$Z_X,\ Z_Y,\ Z_Z\ . \tag{13}$$

The Z_S matrix will contain six independent elements. The diagonal elements will be denoted by

$$Z_{XX},\ Z_{YY},\ Z_{ZZ} \tag{14}$$

and the off-diagonal elements by

$$Z_{XY},\ Z_{XZ},\ Z_{YZ}\ . \tag{15}$$

Orthogonality Correction

Since the direction cosine matrix B defines a transformation of coordinates, it will be an orthogonal matrix. An orthogonal matrix has the property that the product of the matrix and its transpose is the unit matrix

$$BB^T = I\ . \tag{16}$$

The estimated cosine matrix $\hat{B}$ will not remain orthogonal because of computational error. Since Eq. (16) provides a simple check on the orthogonality of a matrix, it is possible to correct the non-orthogonality of $\hat{B}$. The result will be a new estimate of the direction cosines $\hat{B}^*$ which satisfies Eq. (16):

$$\hat{B}^*\hat{B}^{*T} = I.$$

Since an orthogonal matrix has the additional property of transforming a vector without changing its length, it would seem that

the new estimate would be superior to the old estimate for use in Eq. (5) since it will transform the acceleration vector without distorting its magnitude.

The improvement (if any) in performance, because of the orthogonality correction, can be determined once the effect of the correction upon the error (z) matrix is established.

The matrix $\hat{B}$ is contained in the computer as the best estimate of B. Since:

$$\hat{B} = B + E$$
$$\hat{B} = B(I + Z). \tag{16a}$$

Let:

$$T = \frac{1}{2}\left(\hat{B}^T\hat{B} - I\right) ; \tag{17}$$

then

$$T = Z_S + \frac{1}{2} Z^T Z. \tag{18}$$

From Eq. (17) it is obvious that the T matrix will be zero if $\hat{B}$ is orthogonal. If computational error renders the $\hat{B}$ matrix non-orthogonal, then the T matrix will be equal to the Z_S matrix plus a second-order term. Error compensation may be accomplished by including a routine in the airborne computer which ensures that $\hat{B}$ remains orthogonal. Then the T matrix will be driven to zero:

$$T = Z_S + \frac{1}{2} Z^T Z = 0 \tag{19}$$

or

$$Z_{SC} = - \frac{1}{2} Z^T Z$$

where the subscript SC denotes the value of the Z_S matrix after compensation. Since Z_{SC} is a second-order term,

$$Z_{SC} = \frac{1}{2}\left(Z_{SS}^2 - Z_S^2\right) \cong \frac{1}{2} Z_{SS}^2 . \tag{20}$$

By substitution, it is easy to show that

$$\hat{B}^* = \hat{B} - BT \tag{21}$$

provides a new estimated cosine matrix which is orthogonal. Since the true cosine matrix is not known, approximate error compensation may be achieved by

$$\hat{B}^* = \hat{B} - \hat{B}T \tag{22}$$

which differs from Eq. (21) by third-order terms. Non-orthogonality can be completely eliminated by an iterative application of Eq. (22).

Whether or not compensation is done depends upon the particular mission requirements since a trade-off of software complexity versus accuracy is involved.

However, it is clear that algorithms may be evaluated in either a compensated or uncompensated form. The skew-symmetric part of the Z matrix represents a more fundamental limitation since additional information is required to compensate for this error component.

The use of an orthogonality correction will also affect the growth of the direction cosine error which arises during on-board solution of Eq. (2). This aspect of the problem has not been included in previous studies of the direction cosine computation and therefore is treated in some detail in the section on Algorithm Error.

Computational Error

The direction cosine matrix propagates as

$$\dot{B}(t) = \Omega(t)B(t).$$

Unfortunately, for any reasonable vehicle environment the elements of the $\Omega(t)$ matrix will not be known a priori, but must be determined in real time by monitoring the output of the system gyroscopes. Therefore, an analytic solution is not possible in an operational environment and an estimated matrix $\hat{B}$ must be determined by some computational technique. Current methods are based upon the use of standard numerical integration formulas to integrate

the matrix differential equations in real time. The computational error which arises during the integration may be divided into two categories:

> Algorithm error - which is due to the approximate nature of the numerical integration techniques used to integrate Eq. (2). This error would exist even if the computer had infinite word length;
>
> Wordlength error - which is the additional error which arises from the finite computer word length.

For many problems, these two error sources are essentially independent and may be evaluated separately. The total computational error will then be the sum of the two separate errors. Obviously, it would be a poor procedure to minimize one source of error without regard to the other. The first step in achieving a realistic tradeoff between error sources, however, is an accurate assessment of the individual errors.

II. ALGORITHM ERROR

Algorithm error results from the approximate nature of the numerical integration techniques. The actual on-board integration is done in a discrete fashion. If the true direction cosine matrix propagates as

$$B_{K+1} = \Phi_K B_K \tag{23}$$

where Φ_K is the state transition matrix between states K and K+1. Then the approximate cosine matrix propagates as

$$\hat{B}_{K+1} = \hat{\Phi}_K \hat{B}_K \ . \tag{24}$$

Defining

$$\tilde{\Phi} = \hat{\Phi} - \Phi,$$

subtraction gives

$$E_{K+1} = \hat{\Phi}_K E_K + \tilde{\Phi}_K B_K \tag{25}$$

for the step-by-step propagation of the error matrix. The Z matrix is

$$Z_{K+1} = B_{K+1}^{T} E_{K+1} \tag{26}$$

$$Z_{K+1} = B_K^T \Phi_K^T \Phi_K B_K Z_K + B_K^T \Phi_K^T \tilde{\Phi}_K B_K Z_K + B_K^T \Phi_K^T \tilde{\Phi}_K B_K \ .$$

While recognizing the fact that both B and Φ are orthogonal and defining

$$R^* = B_K^T \Phi_K^T \tilde{\Phi}_K B_K \ , \tag{27}$$

then the equation for the Z matrix can be put in the form

$$Z_{K+1} - Z_K = \Delta Z = R^* Z_K + R^* \ . \tag{28}$$

If h is the numerical integration step size (the time interval between steps), then

$$\frac{\Delta Z}{h} = \frac{1}{h}\left(R^* Z_K + R^*\right) \ . \tag{29}$$

The left-hand side now has the form of a difference quotient. The approximation

$$\dot{Z} \cong \frac{\Delta Z}{h} \tag{30}$$

may be introduced. The term ΔZ may be regarded as the per-step error while 1/h is the number of integration steps per unit time, with the derivative being equal to the rate of growth of the Z matrix. Then

$$\dot{Z} = RZ + R \tag{31}$$

where

$$R = R^*/h = f_c R^*$$

and f_c is the computational frequency in steps per second. Eq. (31) is the basic equation which describes the growth of the Z matrix because of the discretization error introduced by the digital computation process. The basic error equations are summarized in Table I.

TABLE I

PROPAGATION OF THE ERROR MATRIX

$$B_{K+1} = \Phi_K B_K$$

$$\hat{B}_{K+1} = \hat{\Phi}_K \hat{B}_K$$

$$\tilde{\Phi} = \hat{\Phi} - \Phi$$

$$R = f_c B^T \Phi^T \tilde{\Phi} B$$

$$\dot{Z} = RZ + R \qquad Z(0) = 0$$

Algorithm Error with an Orthogonality Correction

The determination of the algorithm error is complicated by the use of an orthogonality correction. The orthogonality of the estimated cosine matrix $\hat{B}$ can be tested at any time, and therefore a correction for non-orthogonality can be made at any time. It is entirely feasible to update the estimated cosine matrix $\hat{B}$ for a number of cycles without using an orthogonality correction and to employ the correction routine only periodically to re-establish the orthogonality of the $\hat{B}$ matrix. To a certain extent, this procedure will correct the accumulation of past errors.

The propagation of the Z matrix is given by

$$\dot{Z} = RZ + R \ . \tag{31a}$$

This equation may still be used when an orthogonality correction is included. Suppose that the correction is applied at the times

$$t = nT \qquad n = 1,2,3,....$$

Then, at each one of these times the Z matrix must be re-initialized as

$$Z^*(nT) = Z_{SS}(nT) - \left[\frac{Z^T(nT)\,Z(nT)}{2}\right] . \tag{32}$$

The integration may then be continued to the next correction point.

For the greatest accuracy, the orthogonality correction can be applied everytime the estimated direction cosine matrix $\hat{B}$ is updated. This mode of operation, along with the completely uncompensated mode, provides upper and lower limits for the periodic use of an orthogonality correction. Therefore, it is desirable to determine the propagation of the Z matrix when an orthogonality correction is applied at every step.

After the direction cosine matrix is updated, a correction matrix may be determined as shown previously

$$T = \frac{\hat{B}_{K+1}^T \hat{B}_{K+1} - I}{2} \tag{33}$$

or

$$T = \frac{\left(I + Z_K^T\right) B_K^T \hat{\Phi}_K^T \hat{\Phi}_K B_K \left(I + Z_K\right) - I}{2} \quad .$$

Hereafter the subscript K will be suppressed.

$$T = \frac{\left(I + Z^T\right)\left(I + 2R_S + B^T \tilde{\Phi}^T \tilde{\Phi} B\right)\left(I + Z\right) - I}{2} \tag{34}$$

where

$$2R_S = B^T\left(\Phi^T \tilde{\Phi} + \tilde{\Phi}^T \Phi\right) B \tag{35}$$

is the symmetric part of the R matrix. Neglecting the second-order term in Eq. (34) yields:

$$T = \frac{\left(I + Z^T\right)\left(I + 2R_S\right)\left(I + Z\right) - I}{2} \quad . \tag{36}$$

Interchanging the order of the factors (a second-order effect) gives

$$T = \frac{\left(I + 2Z_S + Z^T Z\right)\left(I + 2R_S\right) - I}{2} \quad . \tag{37}$$

One effect of the T matrix is to replace Z_S by $-Z^T Z/2$ as noted previously. These two values may be considerably different if an orthogonality correction has not been applied recently. However, if a correction is being used everytime the direction cosine matrix is updated, these matrices will be approximately equal in magnitude and

$$2Z_2 + Z^T Z \cong 0 \tag{38}$$

and

$$T \cong R_S. \tag{39}$$

The propagation of the error matrix with a continuous orthogonality correction is therefore

$$\dot{Z}_C = R_{SS} Z_C + R_{SS} \tag{40}$$

where R_{SS} is the skew-symmetric component of the R matrix. This result is summarized in Table II.

TABLE II

PROPAGATION OF THE ERROR MATRIX WITH A CONTINUOUS ORTHOGONALITY CORRECTION

$$\dot{Z}_C = R_{SS} Z_C + R_{SS}$$

$$R_{SS} = \frac{1}{2}\left(R - R^T\right)$$

Closed-Form Solutions

Although the actual vehicle environment is too complex to allow a closed-form solution for the direction cosine matrix, it is a standard error analysis technique to assume an admittedly simplified model to make the analysis more tractable mathematically. Therefore, it will be assumed that the angular rate matrix Ω has a simple form which allows a closed-form solution for the direction cosine matrix (B) to be found. Appendix A shows that it is always possible to take the initial condition of the B matrix as the unit matrix without loss of generality. This initial condition will be implicit in the analysis that follows. The reader is reminded that the significance of the E and Z matrices is due to Eqs. (8) and (9) -- that is, they provide a link between the cosine errors and overall system error. For the time being however, we are only concerned with determining the functional form of the E and Z matrices.

There is one class of closed-form solutions which is easily found. It is well known that the system of matrix equations

$$\dot{X} = A(t)X \qquad X(0) = I \tag{41}$$

has the unique solution

$$X(t) = \exp \int_{t_0}^{t} A(\lambda)\, d\lambda \ , \tag{42}$$

provided that

$$A(t_1)A(t_2) = A(t_2)A(t_1) \tag{43}$$

for all t_1 and t_2. Reference 1 proves that this equality is satisfied if, and only if, the matrix A(t) can be represented by

$$A(t) = \Sigma_i K_i f_i(t) \tag{44}$$

where the f_i are scalar functions and the K_i are mutually commutative constant matrices. For skew-symmetric matrices this condition is satisfied when the constant matrices are proportional. Therefore, the angular rate matrix has the form

$$\Omega(t) = Kf(t) \tag{45}$$

where

$$f(t) = \Sigma f_i(t) \tag{46}$$

The statement "Ω is self-commutative" will be used to indicate that Eq. (43) is satisfied for all t and that the angular rate matrix has the form of Eq. (45). Then, the equation

$$\dot{B} = \Omega B \qquad B(0) = I \tag{47}$$

has the solution

$$B = \exp K \int_{t_0}^{t} f(\lambda)\, d\lambda \tag{48}$$

and if

$$F(t) = \int_{t_0}^{t} f(\lambda)\, d\lambda$$

$$B = \varepsilon^{KF} \tag{49}$$

Next, if

$$K = \begin{bmatrix} 0 & k_z & -k_y \\ -k_z & 0 & k_x \\ k_y & -k_x & 0 \end{bmatrix}$$

and

$$\omega_0^2 = k_x^2 + k_y^2 + k_2^2$$

so that ω_0^2 is the sum of the squares of the elements in the constant matrix K, then

$$K^3 = -\omega_0^2 K$$

$$K^4 = -\omega_0^2 K^2 \tag{50}$$

Using these identities all higher powers of the K matrix may be reduced to either the first or second power. When the exponential function is expanded and like powers of the K matrix are collected, the solution for the B matrix assumes the form

$$B(t) = I + \frac{\sin \omega_0 F(t)}{\omega_0} K + \frac{1 - \cos \omega_0 F(t)}{\omega_0^2} K^2 \qquad (51)$$

This is a closed-form solution for the cosine matrix under the restriction of proportional angular rates about each coordinate axis. To calculate the algorithm error it is necessary to find the appropriate value of the R matrix in Table I. The true state transition matrix for one computational step from t-h to t is given by

$$\Phi(t) = \exp \{KF(t) - KF(t-h)\} , \qquad (52)$$

and if

$$H(t) = F(t) - F(t-h) ,$$

then

$$\Phi(t) = I + \frac{\sin \omega_0 H}{\omega_0} K + \frac{1 - \cos \omega_0 H}{\omega_0^2} K^2 \qquad (53)$$

The R matrix becomes

$$R = f_c \Phi_K^T \tilde{\Phi}_K \qquad (54)$$

because of the commutivity of the matrices. The state transition matrix Φ is given by Eq. (53), while the approximate matrix Φ depends on the numerical integration algorithm which is being evaluated.

Since all powers of the K matrix can be reduced to the first or second power, it follows that the general form of the R matrix is

$$R = r_1 K + r_2 K^2 \qquad (55)$$

where r_1 and r_2 are scalar time functions.

Obviously, the K and K^2 matrices commute, and using the theorem quoted from reference 1, the corresponding solution for

the Z matrix is

$$Z = \varepsilon^{s_1 K} \varepsilon^{s_2 K^2} - I \tag{56}$$

where

$$s_1 = \int_{t_0}^{t} r_1(\lambda)\, d\lambda \tag{57}$$

$$s_2 = \int_{t_0}^{t} r_2(\lambda)\, d\lambda \tag{58}$$

By expanding Eq. (56) and collecting terms, the general form of the Z matrix

$$Z = Z_{SS} + Z_S \tag{59}$$

can be written as

$$Z = z_{SS} K + z_S K^2 \tag{60}$$

where z_{SS} and z_S are the scalar functions

$$z_{SS} = \frac{\varepsilon^{-\omega_0^2 s_2} \sin \omega_0 s_1}{\omega_0} \tag{61}$$

$$z_S = \frac{1 - \varepsilon^{-\omega_0^2 s_2} \cos \omega_0 s_1}{\omega_0^2} .$$

The error relationships are summarized in Table III for the special case of proportional input rates. For this case the nine separate elements of the Z matrix need not be calculated, since the two scalar parameters z_{ss} and z_s completely describe the

TABLE III

PROPORTIONAL RATES

$$\Omega(t) = Kf(t)$$

$$R(t) = r_1(t)K + r_2(t)K^2$$

$$Z = z_{ss}K + z_sK^2$$

$$\alpha = \omega_0 \int_{t_0}^{t} r_1(\lambda)\, d\lambda$$

$$\beta = -\omega_0^2 \int_{t_0}^{t} r_2(\lambda)\, d\lambda$$

$$z_{ss} = \varepsilon^{\beta} \sin \alpha/\omega_0$$

$$z_s = (1 - \varepsilon^{\beta} \cos \alpha)/\omega_0^2$$

the error. As an example the solution for

$$\Omega(t) = K\left(a_1 + a_2t + a_3 \sin \omega_s t\right)$$

is easily found since all the elements of the Ω matrix are proportional. When this restriction is met, an analytical solution for the Z matrix requires only the integration of the R matrix.

When a continuous orthogonality correction is used, the symmetric part of the R matrix is compensated for. The solution for the Z matrix may be modified by setting r_2 (and therefore β) equal to zero. The results are summarized in Table IV. Note that the compensated solution differs from the uncompensated solution by the absence of the exponential term. Whether or not this is of practical importance depends upon the magnitude of the exponential term.

Numerical Integration Formulas

A large number of numerical integration formulas have been proposed for the integration of direction cosines. Since this report is concerned with analysis techniques rather than with a comparative analysis of the different formulas, one or two of the currently used algorithms will be selected for use in examples.

At this point, it will be assumed that the inertial system is instrumented with rate-integrating gyroscopes. These gyroscopes

TABLE IV

PROPORTIONAL RATES
CONTINUOUS ORTHOGONALITY CORRECTION

$\Omega(t) = Kf(t)$	$\alpha = \omega_0 \int_{t_0}^{t} r_1(\lambda)\, d\lambda$
$R_{ss}(t) = r_1(t)K$	$z_{ssc} = \sin \alpha/\omega_0$
$Z_c = z_{ssc}K + z_{sc}K^2$	$z_{sc} = (1 - \cos \alpha)/\omega_0^2$

have an output which is porportional to the integral of the input rate or

$$\theta(t) = \int_{t-h}^{t} \Omega(\lambda)\, d\lambda \tag{62}$$

where $\theta(t)$ is a skew symmetric matrix of gyro outputs.

When

$$\Omega(t) = Kf(t) \ , \tag{63}$$

then

$$\theta(t) = KH(t) \tag{64}$$

and

$$\Phi = \varepsilon^{\theta} \ . \tag{65}$$

Approximate state transition matrices may be generated by expanding the exponential function in a Taylor series and retaining only the higher order terms

$$\varepsilon^{\theta} = I + \theta + \theta^2/2 + \theta^3/6 + \ldots \tag{66}$$

First-Order Taylor Series:

$$\hat{\Phi} = I + \theta \tag{67}$$

Second-Order Taylor Series:

$$\hat{\Phi} = I + \theta + \theta^2/2 \tag{68}$$

Note that these approximations do not assume that the angular rate matrix Ω is constant during an integration step. The basic assumption is that Ω is self-commutative. There is, of course, the additional judgment that the number of terms retained is sufficiently accurate for the application at hand.

Narrow-band Solution

Although it is possible to determine an exact form of the R matrix for many representative rate matrices, it is much simpler to use an approximate solution which is adequate for almost any practical situation. For this reason, exact solutions will be discussed in Appendix B, while the remainder of this section will consider an approximate solution.

Given the state transition matrix

$$\Phi = I + \frac{\sin \omega_0 H}{\omega_0} K + \frac{1 - \cos \omega_0 H}{\omega_0^2} K^2 \tag{69}$$

where

$$H = F(t) - F(t-h) \ ,$$

a narrow-band solution may be defined for the situation where

$$\omega_0 H \leq 0.5 \ . \tag{70}$$

With this restriction

$$\sin \omega_0 H \cong \omega_0 H - \frac{\omega_0^3 H^3}{6}$$

$$\cos \omega_0 H \cong 1 - \frac{\omega_0^2 H^2}{2} + \frac{\omega_0^4 H^4}{24} .$$

When the equality of Eq. (70) is satisfied, the magnitude of the first neglected term is only one tenth that of the fourth-order term which is retained. The term "narrow-band" is applied since the functions sin $\omega_0 H$ and cos $\omega_0 H$ also arise in the study of narrow-band FM signals. Then

$$\Phi \cong I + \left(H - \omega_0^2 \frac{H^3}{6}\right)K + \left(\frac{H^2}{2} - \omega_0^2 \frac{H^4}{24}\right)K^2 . \tag{71}$$

It is now a simple matter to determine on approximate R matrix for the Taylor Series algorithms.

First-Order Taylor Series

$$\hat{\Phi} = I + \theta = I + HK \tag{72}$$

$$R = f_c \Phi^T \tilde{\Phi}$$

$$R = -\left(f_c \omega_0^2 H^3/6\right)K - \left(f_c H^2/2\right)K^2$$

Second-Order Taylor Series

$$\hat{\Phi} = I + \theta + \frac{\theta^2}{2} = I + HK + \frac{H^2}{2} K^2 \tag{73}$$

$$R = \left(f_c \omega_0^2 H^3/6\right)K - \left(f_c \omega_0^2 H^2/8\right)K^2 .$$

From the mean value theorem of differential calculus:

$$H(t) = F(t) - F(t-h) = hf(t-\eta h)$$

where

$$0 \leq \eta \leq 1 .$$

Therefore:

$$H(t) = hf(t - \eta h) \ .$$

The mean value of f(t) may be approximated by f(t) itself since the small shift in the time axis of the error solution is not significant. The final form of the R matrices for the Taylor Series algorithm are listed in Table V.

TABLE V

TAYLOR SERIES NARROWBAND SOLUTION

1st Order	$\left(-\omega_0^2/3f_c^2\right)f^3(t)K - \left(1/2f_c\right)f^2(t)K^2$
2nd Order	$\left(\omega_0^2/6f_c^2\right)f^3(t)K - \left(\omega_0^2/8f_c^3\right)f^4(t)K^2$

Table V then provides a value of the R matrix for any arbitrary rate input, provided that the rate matrix $\Omega(t)$ is self-commutative, and provided that the narrow-band restriction

$$\omega_0 hf(t) \leq 0.5$$

is satisfied.

Note that the narrow-band criteria are <u>not</u> dependent upon the frequency content of the rate matrix, but rather upon its amplitude. For example, the rate inputs

$$\omega_x = \omega_y = \omega_z = \frac{10}{\sqrt{3}} \cos t \qquad (74)$$

$$\omega_x = \omega_y = \omega_z = \frac{10}{\sqrt{3}} \cos 10t$$

both have the same amplitude with

$$\omega_0 = 10 \ ,$$

although their frequency content differs by a factor of 10. If the computational rate is 20 times a second or greater

$$\omega_0 H \leq 0.5$$

and both inputs may be categorized as narrow-band. For the 20 sample/sec situation, the inaccuracy in the error calculation would be approximately 10 percent. However, for any application with such high amplitude input rates the sampling frequency would, in all probability, exceed 20 samples/sec. This would reduce the inaccuracy in the error calculation. The narrow-band criteria will adequately represent almost any practical situation.

Example I--Constant Rate

Let

$$f(t) = 1$$

$$\Omega(t) = K \ .$$

Then

$$H(t) = h \ .$$

From Table V, the R matrices are

First-Order:

$$R = -\left(\omega_0^2/3f_c^2\right)K - \left(1/2f_c\right)K^2 \tag{75}$$

Second-Order:

$$R = \left(\omega_0^2/6f_c^2\right)K - \left(\omega_0^2/8f_c^3\right)K^2 \ . \tag{76}$$

In both cases the R matrix is a constant and the integration called for in Table V is easily performed. The results appear as Table VI.

TABLE VI

CONSTANT RATE NARROW-BAND SOLUTION
TAYLOR SERIES ALGORITHMS

	α	β
1st Order	$-\left(\omega_0^3/3f_c^2\right)t$	$\left(\omega_0^2/2f_c\right)t$
2nd Order	$\left(\omega_0^3/6f_c^2\right)t$	$\left(\omega_0^4/8f_c^3\right)t$

Example II--Sinusoidal Rate

Let

$$f(t) = \cos \omega_s t \ .$$

Then

$$H(t) = \frac{2}{\omega_s} \sin\left(\frac{h\omega_s}{2}\right) \cos\left[\omega_s\left(t - \frac{h}{2}\right)\right] \tag{77}$$

which for small h is approximately

$$H(t) \cong h \cos \omega_s\left(t - \frac{h}{2}\right) , \tag{78}$$

so that

$$\eta \cong \omega_s \frac{h}{2}$$

which may be approximated by

$$H(t) = h\cos \omega_s t \ .$$

The R matrices are then

First-Order:

$$R = -\left(\omega_0^2/3f_c^2\right) \cos^3 \omega_s t\, K - \left(1/2f_c\right) \cos^2 \omega_s t\, K^2 \qquad (79)$$

Second-Order

$$R = \left(\omega_0^2/6f_c^2\right) \cos^3 \omega_s t\, K - \left(\omega_0^2/8f_c^3\right) \cos^4 \omega_s t\, K^2 \qquad (80)$$

These expressions for the R matrices must now be integrated in accordance with Table III to obtain the error terms z_{ss} and z_s. The results are given in Table VII.

TABLE VII

SINUSOIDAL RATE NARROWBARD SOLUTION
TAYLOR SERIES ALGORITHMS

1st Order	α	$-\left(\omega_0^3/3f_c^2\right)\left(\frac{\sin \omega_s t}{\omega_s} - \frac{\sin^3 \omega_s t}{3\omega_s}\right)$
	β	$\left(\omega_0^2/2f_c\right)\left(\frac{t}{2} + \frac{\sin 2\omega_s t}{4\omega_s}\right)$
2nd Order	α	$\left(\omega_0^2/6f_c^2\right)\left(\frac{\sin \omega_s t}{\omega_s} - \frac{\sin^3 \omega_s t}{3\omega_s}\right)$
	β	$\left(\omega_0^4/8f_c^3\right)\left(\frac{3}{8}t + \frac{\sin 2\omega_s t}{4\omega_s} + \frac{\sin 4\omega_s t}{32\omega_s}\right)$

For both the sinusoidal and constant rate cases the total error is

$$Z = \varepsilon^\beta \frac{\sin \alpha}{\omega_0} K + \frac{1 - \varepsilon^\beta \cos \alpha}{\omega_0^2} K^2 \ . \qquad (81)$$

When a continuous orthogonality correction is used, the error becomes

$$Z_c = \frac{\sin \alpha}{\omega_0} K + \frac{1 - \cos \alpha}{\omega_0^2} K^2 \tag{82}$$

Simulation

A solution for the Z matrix can often be found by simulation. The algorithm under study is programmed on a digital computer and used to integrate a specified $\theta(t)$ for which an exact cosine matrix is known. The exact solution is then subtracted from the simulation result to determine the error matrix (see Figure 2). In this report the term "simulation" will always be used in this context. Its purpose will be to provide a check on the analytical results.

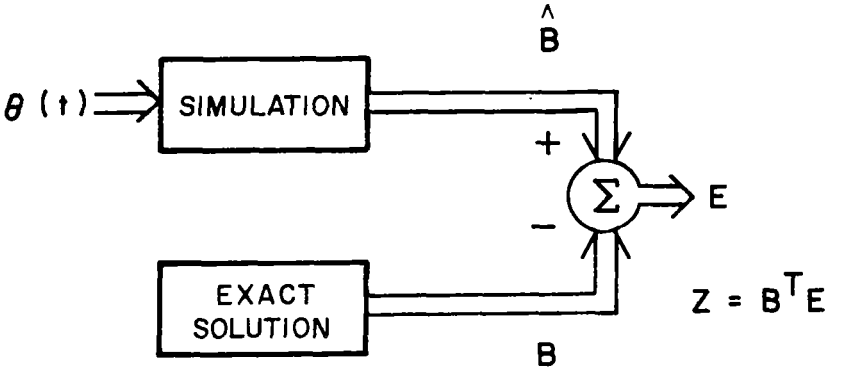


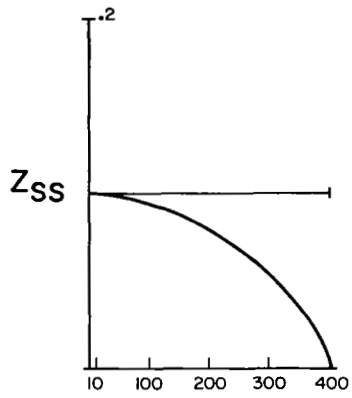
Figure 2.- Effect of direction cosine error

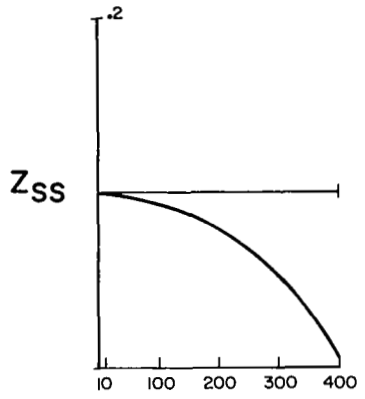
Simulations were conducted for both sinusoidal and constant rate inputs. The results are shown in Figures 3 through 6. Parameter values for the simulations are given in Table VIII. The error was calculated for 40 or 400 seconds. The figures compare the simulation results to a plot of the analytical solution obtained using the narrow-band approximation. These cases are repeated with a continuous orthogonality correction in Figures 7 through 10.

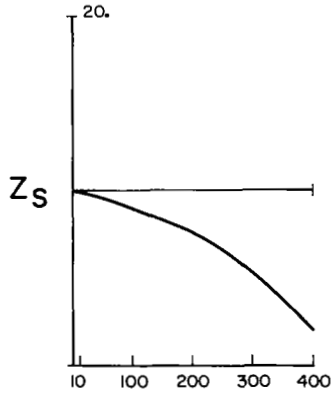
TABLE VIII

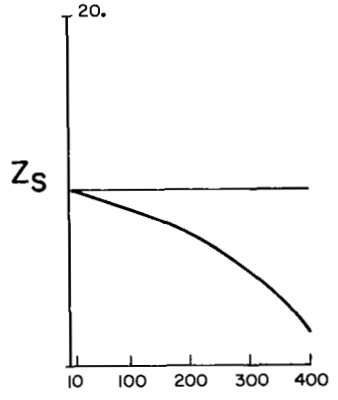
SIMULATION PARAMETER VALUES

$\omega_x = 0.3$	$f_c = 32$
$\omega_y = 0.3$	$\omega_s = 0.25$
$\omega_z = 0.3$	





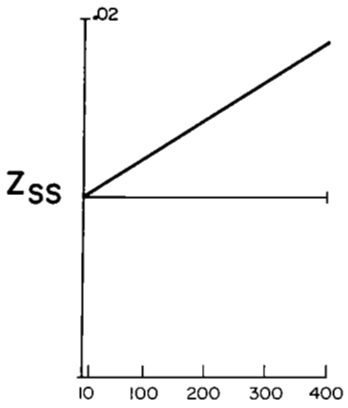


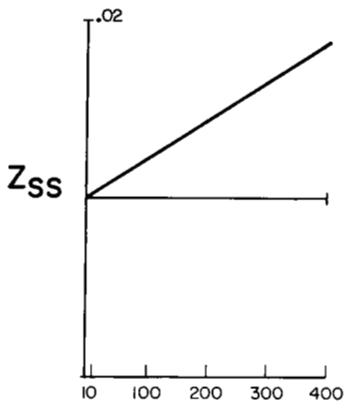


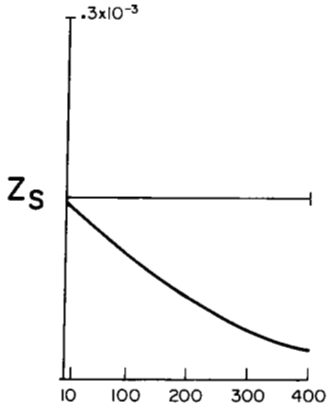
SIMULATION RESULT

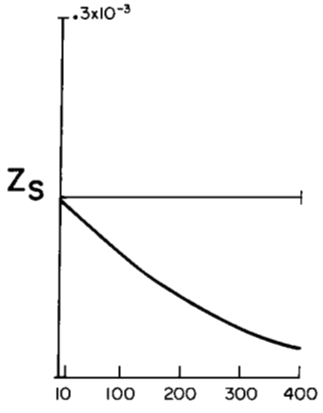
ANALYTICAL SOLUTION

Figure 3.- Constant rates, first-order Taylor series









SIMULATION RESULT ANALYTICAL SOLUTION

Figure 4.- Constant rates, second-order Taylor series

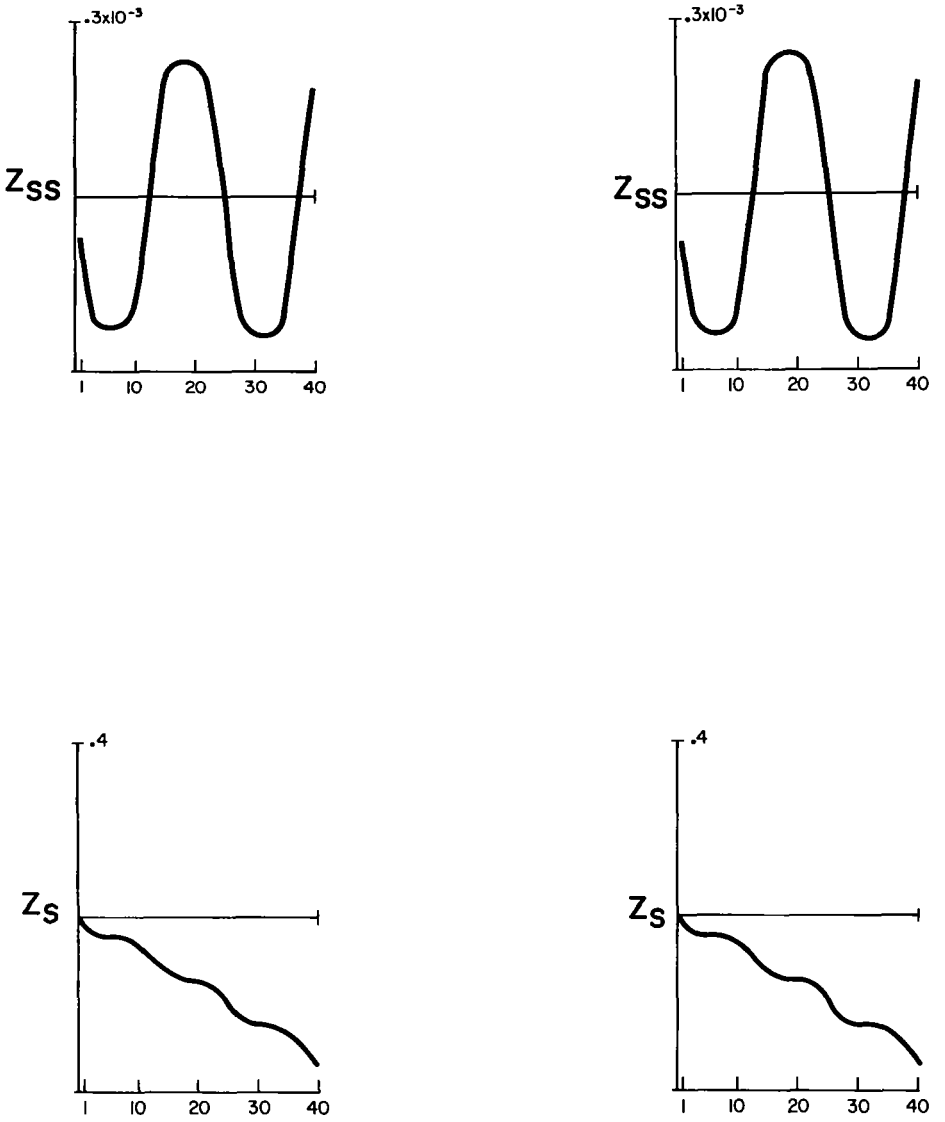


Figure 5.- Sinusoidal rates, first-order Taylor series

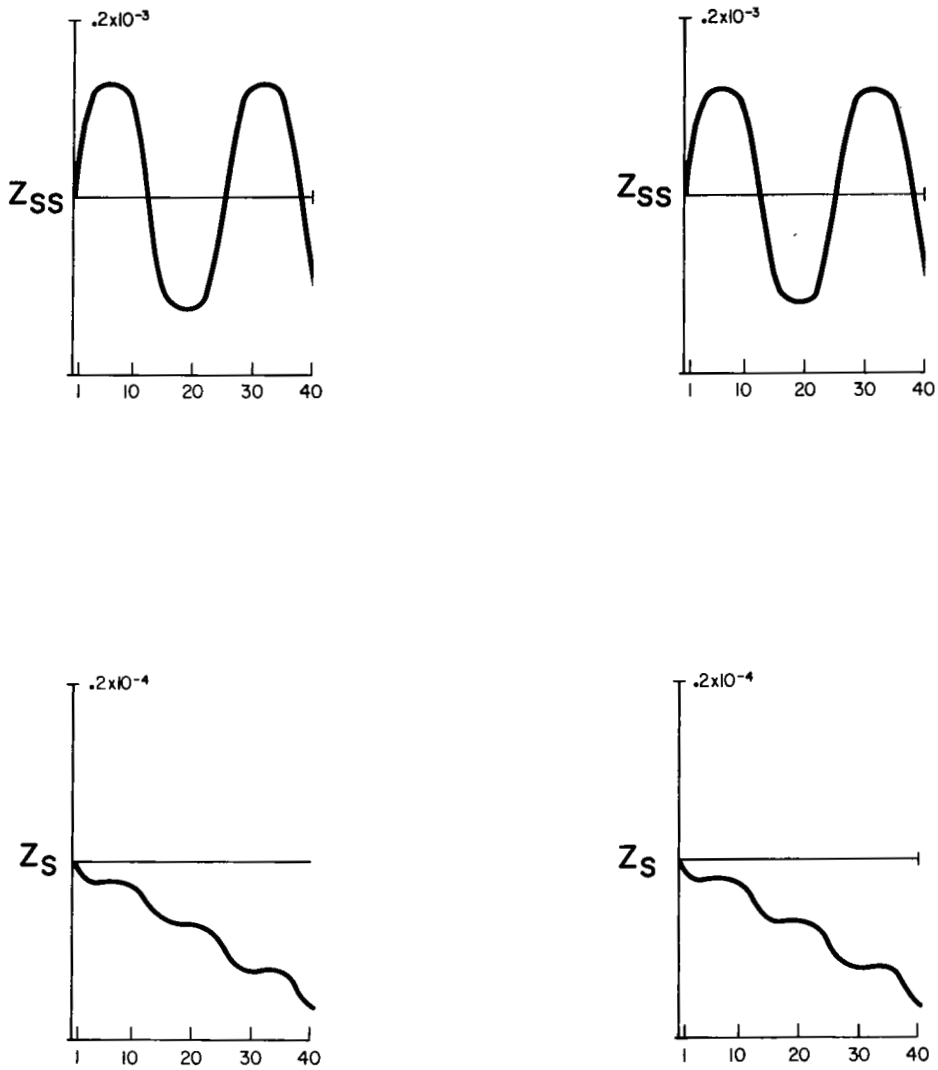


Figure 6.- Sinusoidal rates, second-order Taylor series

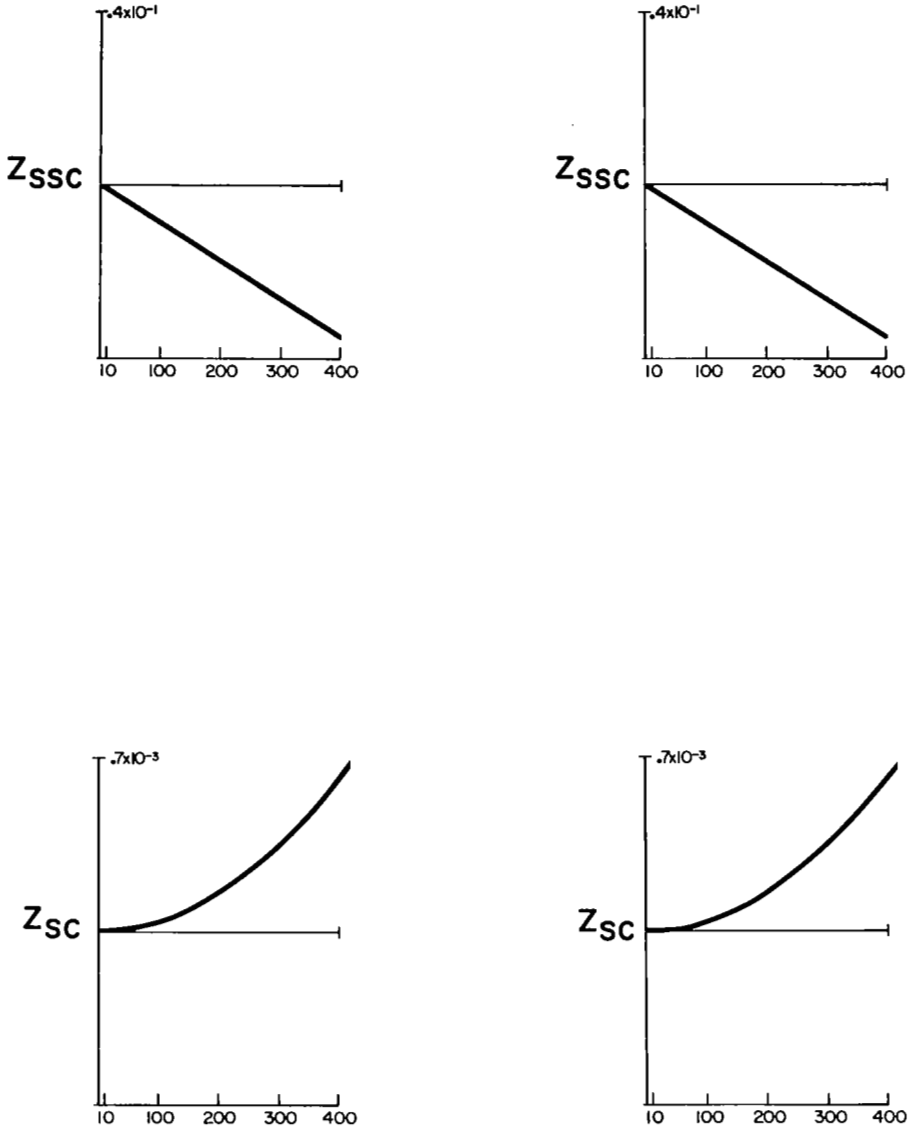
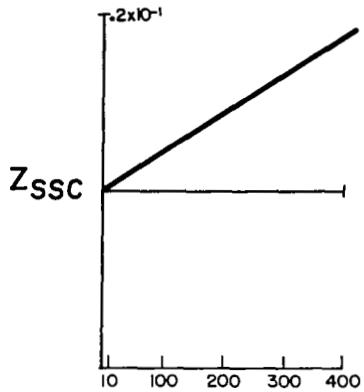
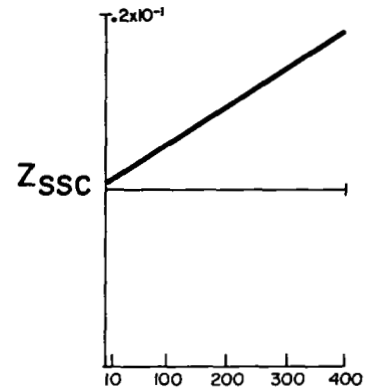
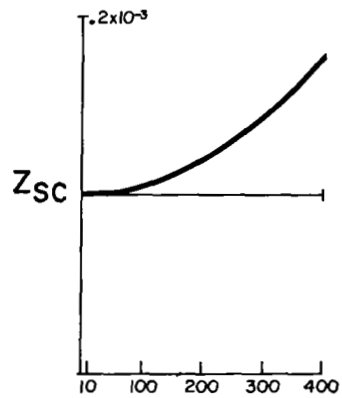
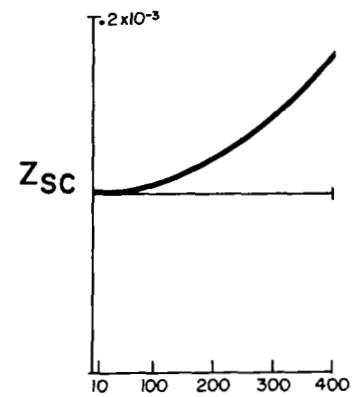


Figure 7.- Constant rates, first-order Taylor series, continuous orthogonality correction









SIMULATION RESULT

ANALYTICAL SOLUTION

Figure 8.- Constant rates, second-order Taylor,series, continuous orthogonality correction

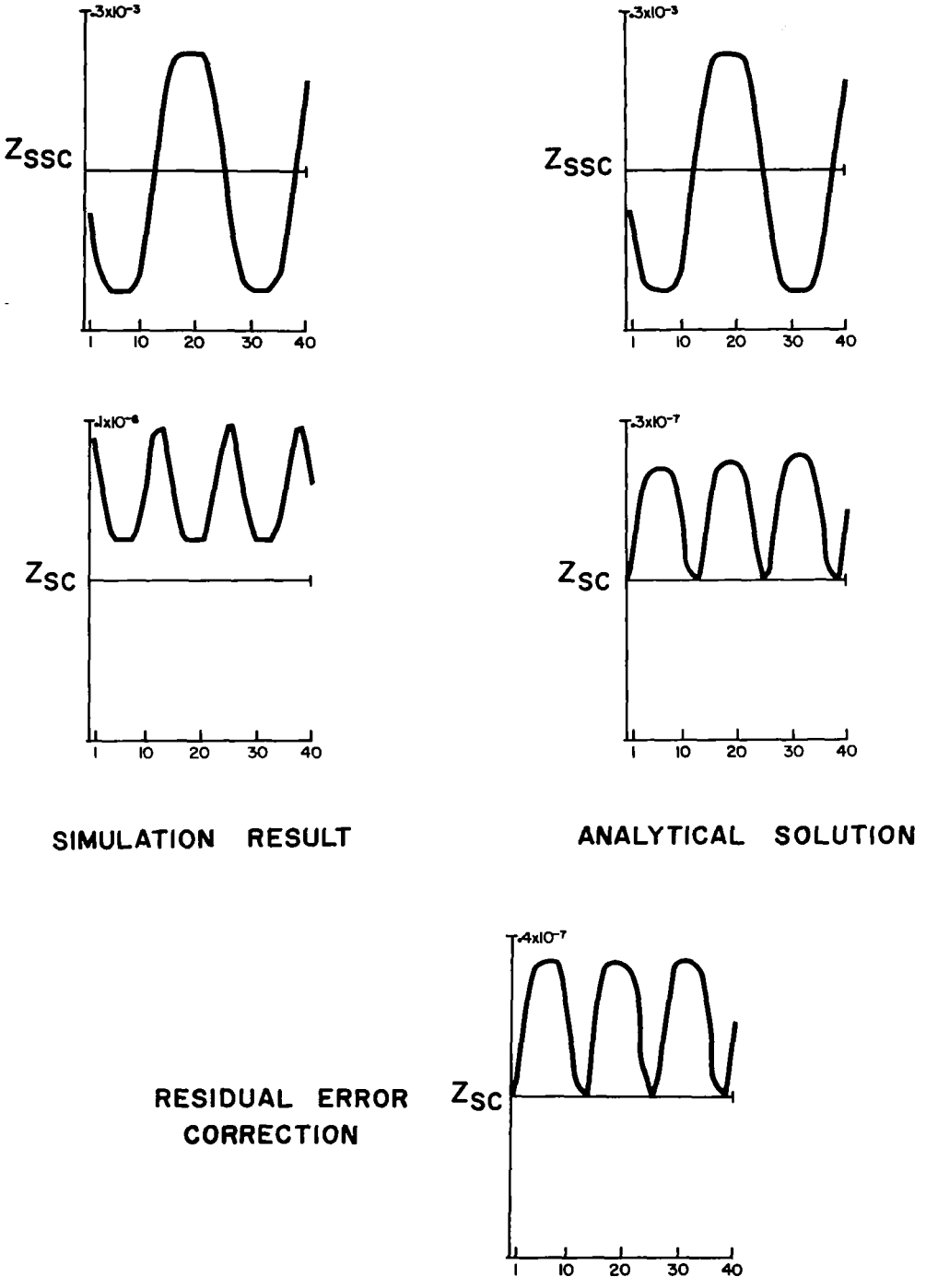


Figure 9.- Sinusoidal rates, first-order Taylor series, continuous orthogonality correction

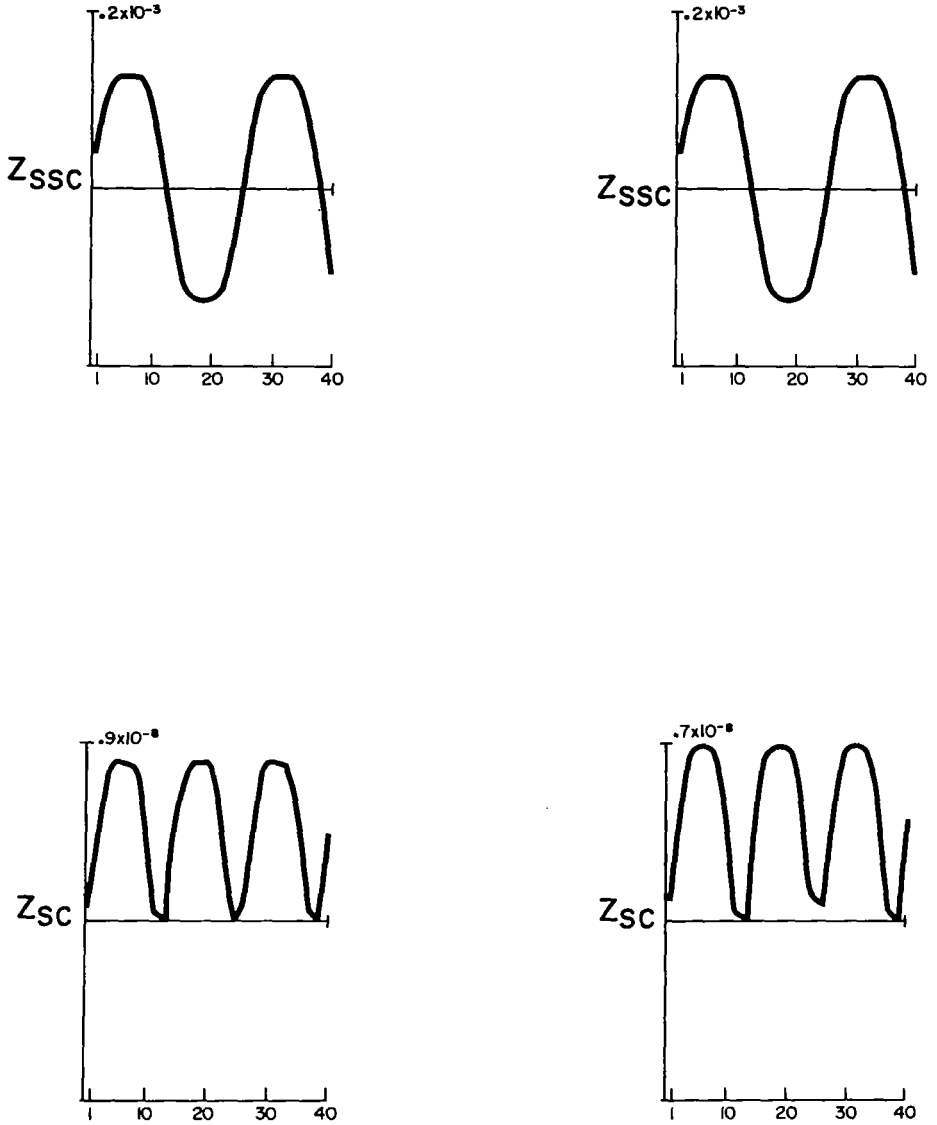


Figure 10.- Sinusoidal rates, second-order Taylor series, continuous orthogonality correction

The difference between the analytical and simulation results of Figure 9 is of interest. In deriving the differential equation (Table II) for the growth of the Z matrix with an orthogonal correction, it was assumed that

$$2Z_s + Z^T Z = 0 \; ; \tag{83}$$

that is, with the correction continuously applied the $\hat{B}$ matrix will remain orthogonal. The compensated Z matrix is then

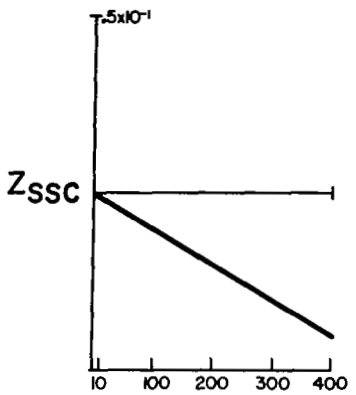
$$Z_c = z_{ssc}K + z_{sc}K^2 \tag{84}$$

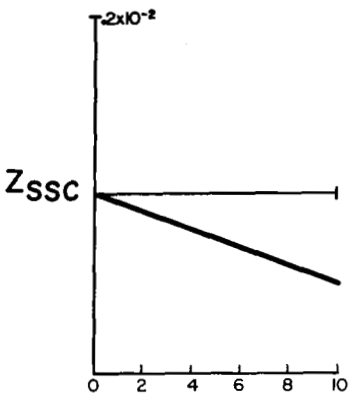
for which Eq. (83) is satisfied. However, this approximation will be in error by the non-orthogonality error introduced since the last orthogonality correction. More accurately

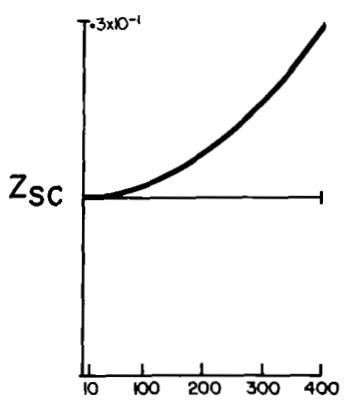
$$Z_c = z_{ssc}K + (z_{sc} + z_p)K^2 \tag{85}$$

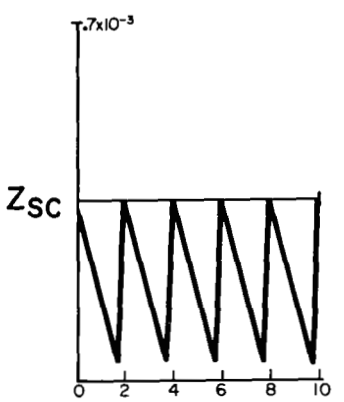
where z_p is the error introduced since the last correction. When the orthogonality correction is applied z_p will be compensated. However, it will generate a second-order residual because of the Z^TZ term. This residual error results from the incomplete compensation of the symmetric component of the Z matrix by the orthogonality correction. When an orthogonality correction is applied at each integration interval, z_p will be equal to the symmetric error introduced in a single step. From Table V it is apparent that while this error may be appreciable for a first-order algorithm, it will be much smaller for the second-order algorithm. The residual error is completely insignificant in the simulation results of Figure 10 for the second order algorithm.

The residual error is not serious because it is not propagated. Each time the orthogonality correction is applied it compensates for the old error and introduces a new residual error. The new error will depend on the amplitude of the angular rates over the integration interval. The residual error can be reduced for the first-order algorithm by decreasing the integration interval, and thereby reducing the per step error. It can be eliminated by applying the orthogonality correction twice per integration step. The second application corrects for the residual error of the first. Figure 9 contains a plot of the symmetric error term when an orthogonality correction is used twice each integration step to correct for the residual error.









SIMULATION RESULT

SIMULATION RESULT

Figure 11.- Constant rates, first-order Taylor series, periodic orthogonality correction

For the angular rates, integration interval and simulation times are chosen, and the parameters α and β remain small. The solutions of Tables II and III may be approximated by

$$z_{ss} \cong \frac{\alpha}{\omega_0}(1 + \beta) \tag{86}$$

$$z_{ssc} \cong \frac{\alpha}{\omega_0}$$

$$z_s \cong \frac{-\beta}{\omega_0^2} + \frac{\alpha^2}{2\omega_0^2}(1 + \beta)$$

$$z_{sc} \cong \frac{\alpha^2}{2\omega_0^2} .$$

The difference between the compensated and uncompensated solutions is due to the parameter β which, in turn, is proportional to the symmetric component of the R matrix. The first-order algorithm is characterized by a large symmetric error component. It is therefore useful for short-term missions which involve low angular rates. The effect of β on the accuracy of a first-order algorithm is evident when a comparison is made between Figures 3 and 7.

A more interesting comparison might be made between a compensated first-order algorithm and an uncompensated second-order algorithm. At the same computational frequency both should require roughly equal computer time and memory. A sinusoidal rate about one or more axes can be used to approximate vehicle limit cycling or angular vibrations. As illustrated by Figures 6 and 9, the first-order algorithm would have a skew-symmetric error twice that of the second-order algorithm. This error is oscillatory. The uncompensated second-order algorithm, on the other hand, has a secular symmetric error term, while the symmetric error in the first-order algorithm is bounded. The choice of a "best" algorithm is not always simple since it involves a trade-off between the allowable error, the expected environment, and the computational resources available. A subsequent section will consider this trade-off in more detail.

In Figures 7 through 10, the orthogonality correction was applied at each integration step or 32 times a second. It is possible to apply the correction much less frequently. In Figure 11 an orthogonality correction is periodically employed.

A first-order algorithm with a constant input rate was used. The correction was applied every 64 integration steps (once every 2 seconds). Since the residual error of a first order algorithm would be quite large after 64 steps, the correction was applied twice at the 2-second intervals. By comparing Figures 3 and 11 it is evident that even the infrequent use of the correction prevented the exponential increase in the skew-symmetric error term.

Figure 11 actually provides a plot of the symmetric error immediately after a correction has been applied. (The error is plotted at 10-second intervals over a total time of 400 seconds.) Additional plots in Figure 11 show the error for the first 10 seconds at intervals of 0.25 second. Note that the error grows in an uncompensated manner between the 2-second compensation points. Therefore:

1. Figure 3 represents the uncompensated error,

2. Figure 7 represents the continuously compensated error,

3. The 400-second plot of Figure 11 is somewhat misleading since it represents the symmetric error immediately after an orthogonality correction. This discrepancy may be resolved by also plotting the error immediately before the orthogonality correction. The effective error may then be considered to be the average of the "before" and "after" values. The effective error is then directly dependent upon the frequency at which the correction is applied. This frequency is another trade-off parameter between computer loading and error level.

General Solution

When the angular rate matrix is self-commutative, i.e.:

$$\Omega(t_1)\Omega(t_2) = \Omega(t_2)\Omega(t_1) \tag{87}$$

for all t and R matrix is immediately known (Table V). In addition, if $\Omega(t)$ can be expressed as the sum of simple analytical functions, it is usually possible to integrate the R matrix and find the total analytical solution for the algorithm error (Table III). In general, however, $\Omega(t)$ will have the form:

$$\Omega(t) = \Sigma K_i f_i \tag{88}$$

where the K_i matrices are not mutually commutative. In this case, a complete analysis is much more difficult, since no analytical

solution for the direction cosines is known. The well known Neumann series does provide a symbolic solution, however:

$$B(t) = I + \int_{t_0}^{t} \Omega(\lambda)\, d\lambda + \int_{t_0}^{t}\int_{t_0}^{t} \Omega(\lambda)\Omega(\tau)\, d\tau d\lambda + \ldots \qquad (89)$$

Therefore:

$$\Phi(t) = I + \int_{t-h}^{t} \Omega(\lambda)\, d\lambda + \int_{t-h}^{t}\int_{t-h}^{\lambda} \Omega(\lambda)\Omega(\tau)\, d\tau d\lambda + \ldots \qquad (90)$$

from Eq. (62)

$$\theta(t) = \int_{t-h}^{t} \Omega(\lambda)\, d\lambda \ .$$

Now define:

$$\theta_n(t) = \int_{t-h}^{t} \Omega(\lambda)\theta_{n-1}(\lambda)\, d\lambda \qquad \theta_0(t) = I \ ; \qquad (91)$$

then

$$\Phi = I + \theta + \sum_{n=2}^{\infty} \int_{t-h}^{t} \Omega\theta_{n-1}\, d\lambda$$

This series may be written as

$$\Phi = I + \theta + \sum_{n=2}^{\infty} \int_{t-h}^{t} \left\{ \frac{1}{n!} \frac{d\theta^n}{dt} + \Omega\theta_{n-1} - \frac{1}{n!} \frac{d\theta^n}{dt} \right\} d\lambda \qquad (92)$$

or

$$\Phi = \varepsilon^{\theta} + \sum_{n=2}^{\infty} C_n^* \tag{93}$$

where

$$C_n^* = \int_{t-h}^{t} \left\{ \Omega\theta_{n-1} - \frac{1}{n!} \frac{d\theta^n}{dt} \right\} d\lambda \ . \tag{94}$$

Of course, Eq. (93) is no simpler than the original Eq. (89). Actually, it is more complex. However, Eq. (93) provides a link with previous error estimates. If the angular rate matrix is self-commutative, the right-hand side of Eq. (93) reduces to the exponential function alone. The series in C_n^* may thus be interpreted as the additional error resulting from a non-commutative rate matrix. The exponential function may be written as

$$\varepsilon^{\theta} = I + \frac{\sin \phi(t)}{\phi(t)} \theta(t) + \frac{1 - \cos \phi(t)}{\phi^2(t)} \theta^2(t) \tag{95}$$

where $\phi^2(t)$ is a scalar function equal to the sum of the squares of the three independent elements of the matrix $\theta(t)$. Eq. (95) is a generalized form of Eq. (51) which represented the exponential function for the case of self-commutative rate matrix. The error due to the approximation of the exponential term may be determined by the same procedure which was applied to the earlier situation. Since

$$R = f_c B^T \Phi^T \tilde{\Phi} B \ , \tag{96}$$

let

$$W = f_c \Phi^T \tilde{\Phi} = f_c (\tilde{\Phi} - \theta \tilde{\Phi}) \ .$$

Once again

$$\theta(t) = h\Omega(t - \eta h)$$

$$0 \leq \eta \leq 1$$

by the mean value theorem. To simplify the determination of the narrow-band approximation may be used. The result is presented in Table IX:

TABLE IX

TAYLOR SERIES NARROWBAND SOLUTION
GENERAL RATE MATRIX

Order	W_{NB}
First	$\frac{-\omega_0^2(t)\Omega(t)}{3f_c^2} - \frac{1}{2f_c}\Omega^2(t) - C_2(t) - C_3(t)$
Second	$\frac{\omega_0^2(t)\Omega(t)}{6f_c^2} - \frac{\omega_0^2\Omega^2(t)}{8f_c^3} - C_2(t) - C_3(t)$

Table IX is a generalization of Table V for the case where $\Omega(t)$ is not necessarily self-commutative. The two matrices C_2 and C_3 represent the additional error which results from the non-commutativity of the angular rate matrix. This error is commonly called the "commutatively" error. From Eq. (96):

$$C_2 = f_c C_2^*$$

$$C_3 = f_c(C_3^* - \theta C_2^*)$$

From Eq. (94):

$$C_2^* = \frac{1}{2}\int_{t-h}^{t} \Omega(\lambda)\theta(\lambda) - \theta(\lambda)\Omega(\lambda)\, d\lambda \tag{97}$$

$$C_3^* = \int_{t-h}^{t} \Omega(\lambda)\theta(\lambda) - \left[\theta^2(\lambda)\Omega(\lambda) + \theta(\lambda)\Omega(\lambda)\theta(\lambda) + \Omega(\lambda)\tilde{\theta}(\lambda)/6\right] d\lambda \tag{98}$$

Since θ and Ω are skew-symmetric, the principal commutativity term C_2^* will be also. All three matrices have three independent elements. Eq. (97) may also be written in terms of a vector cross-product

$$\underline{c}_2^* = \frac{1}{2}\int_{t-h}^{t} \underline{\theta}\times\underline{\omega}\, d\lambda \tag{99}$$

where each vector contains the three independent elements of the corresponding matrix. This equation allows another interpretation of the major commutativity error term; it results when the vectors θ and ω are not co-linear during the integration interval. This implies that the direction of the angular rate vector is changing with time.

Although a non-commutative rate matrix considerably complicates an error analysis, it is, unfortunately, the rule rather than the exception. The solution for a self-commutative rate matrix is very useful for a comparative analysis of different direction cosine algorithms; however, when a more realistic environment is assumed to determine system performance, non-commutativity errors inevitably arise. In these circumstances, it is necessary to evaluate their contribution to the total error.

Example I

The trajectory of a launch vehicle may often be described by a series of constant rate turns. It follows that an analytical solution is known for the direction cosine matrix. Since the turning rate is generally small, almost any numerical integration algorithm will demonstrate very small errors. However, a more

realistic system performance analysis might include the addition of sinusoidal rates about one, or more, axis to represent angular vibrations or vehicle limit cycles. For simplicity, consider only one sinusoidal term:

$$\Omega(t) = K_0 + K_1 \cos \omega_s t \tag{100}$$

Most likely, the matrices K_0 and K_1 will not commute and a commutativity error will result. Let:

$$L = K_1 K_0 - K_0 K_1 \tag{101}$$

The commutativity term of most interest is C_2 since it is of highest order and being skew-symmetric will not be reduced by an orthogonality correction. A direct evaluation of Eq. (97) gives:

$$C_2(t) = \frac{f_c}{2} L(P_3 \cos \omega_s t + P_2 \sin \omega_s t) \tag{102}$$

where

$$P_2 = \frac{2 \sin \omega_s h}{\omega_s^2} - \frac{h(1 + \cos \omega_s h)}{\omega_s} \tag{103}$$

$$P_3 = \frac{h \cos \omega_s h}{\omega_s} - \frac{2(1 - \cos \omega_s h)}{\omega_s^2}$$

If:

$$\omega_s h \ll 1$$

the C_2 matrix is approximately

$$C_2(t) = \frac{\omega_s L}{2 f_c^2} \sin \omega_s t \ . \tag{104}$$

In this case, the commutativity error term is sinusoidal as might be expected from the interaction of a constant rate and a sinusoidal rate.

Example II

The so-called "generalized coning motion" is defined by an angular rate matrix, the elements of which are

$$\begin{aligned} \omega_x(t) &= \omega_{x0} \\ \omega_y(t) &= \omega_{y0} \cos \omega_s t \\ \omega_z(t) &= \omega_{z0} \sin \omega_s t. \end{aligned} \tag{105}$$

The principal commutativity term is most easily found from Eq. (99). If the error vector is denoted by

$$\underline{c}_2 = \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix}, \tag{106}$$

then

$$c_x = \frac{\omega_{y0}\omega_{z0}}{2} f_c P_1 \tag{107}$$

$$c_y = \frac{\omega_{x0}\omega_{z0}}{2} f_c (P_2 \cos \omega_s t + P_3 \sin \omega_s t)$$

$$c_z = \frac{\omega_{x0}\omega_{y0}}{2} f_c (-P_2 \sin \omega_s t + P_3 \cos \omega_s t)$$

where

$$P_1 = \frac{h}{\omega_s} - \frac{\sin \omega_s h}{\omega_s^2} \tag{108}$$

and P_2 and P_3 are defined in Example I.

Again, when $\omega_s h \ll 1$

$$c_x \cong \frac{\omega_{y0}\omega_{z0}\omega_s}{12f_c^2} \tag{109}$$

$$c_y \cong \frac{\omega_{x0}\omega_{z0}\omega_s}{12f_c^2} \cos \omega_s t$$

$$c_z \cong \frac{\omega_{x0}\omega_{y0}\omega_s}{12f_c^2} \sin \omega_s t$$

The x axis commutativity term is a constant. A constant term in the R matrix will cause a secular term in the Z matrix (see Table I). This secular term may greatly degrade overall system performance. The commutativity terms for the y and z axes, being periodic, are much less important.

Example III

To determine a general form for the commutativity error, note that, by the mean value theorem:

$$\Omega(\lambda) = \Omega(t_0) + \dot{\Omega}(t_1)\lambda \tag{110}$$

where

$$t_0 = t-h \leq t_1 \leq t ,$$

$$0 \leq \lambda \leq h$$

although the theorem does not provide the value of t_1.

Let

$$\Omega_0 = \Omega(t_0) \tag{111}$$

$$\dot{\Omega}_1 = \dot{\Omega}(t_1) .$$

A straightforward evaluation of Eq. (97) gives

$$C_2^* = \frac{1}{12f_c^3}\left(\dot{\Omega}_1\Omega_0 - \Omega_0\dot{\Omega}_1\right) \tag{112}$$

$$C_3^* = \frac{1}{24f_c^4}\left(\dot{\Omega}_1\Omega_0^2 - \Omega_0^2\dot{\Omega}_1\right)$$

and

$$C_2 = f_c C_2^* = \frac{1}{12f_c^2}\left(\dot{\Omega}_1\Omega_0 - \Omega_0\dot{\Omega}_1\right) \tag{113}$$

$$C_3 = \frac{1}{24f_c^3}\left(\dot{\Omega}_1\Omega_0^2 - 2\Omega_0\dot{\Omega}_1\Omega_0 + \Omega_0^2\dot{\Omega}_1\right)$$

It follows that both C_2 and C_3 are skew-symmetric. Since C_3 will usually be much smaller than C_2, it may be neglected in comparison to C_2. For both first- and second-order Taylor series algorithms, the commutativity error may be considered to be skew-symmetric and inversely proportional to f_c^2.

Once the commutativity error matrix is found, the W matrix is easily obtained from Table IX. Unfortunately, even if the W matrix is easily integrable, a closed-form solution for the Z matrix is not available as it was in the case of a self-commutative rate matrix. However, a solution may always be computed by numerical integration techniques. A procedure for determining the Z matrix once the W matrix is known is outlined in the section on Performance Analysis. It will not be pursued further because a subsequent section will demonstrate that a computable solution for the Z matrix which does not require the determination of the W matrix is possible.

Compensation Technique

Given the basic error equation [Eq. (31)]:

$$\dot{Z} = RZ + R$$

the R matrix can be expressed as the sum of a symmetric and a skew-symmetric matrix:

$$R = R_{ss} + R_s \quad . \tag{114}$$

For a short time, or while Z is small, the initial error growth can be approximated by

$$\dot{Z} = R \tag{115}$$

if an orthogonality correction is not used, and by

$$\dot{Z}_{ssc} = R_{ss} \tag{116}$$

$$\dot{Z}_{sc} = R_{ss} Z_{ssc}$$

when the correction is used. Continuing with the case of no orthogonality correction, let

$$S = \int_0^t R \, d\lambda = S_{ss} + S_s \quad . \tag{117}$$

If B_A is calculated using a second-order algorithm, the initial error growth will be

$$B_A = B(I + S_{ss} + S_s) \tag{118}$$

where S_{ss} is inversely proportional to f_c^2 and S_s is inversely proportional to f_c^3. Assume that B_B is calculated with the same algorithm but with a computational rate one half that used to calculate B_A.

Then

$$B_B = B(I + 4S_{ss} + 8S_s) \quad . \tag{119}$$

These two relationships can now be used to eliminate either S_{ss} or S_s.

For example:

$$B_C = \frac{4B_A - B_B}{3} = B\left(\frac{I - 4S_s}{3}\right). \tag{120}$$

The skew-symmetric term is compensated at the cost of an increase in the symmetric term. This term, however, is often much smaller than the skew-symmetric term. A corresponding correction for a first-order algorithm is

$$B_C = \frac{4B_A - B_B}{3} = B\left(\frac{I + 2S_s}{3}\right) \tag{121}$$

which provides a reduction in the symmetric term as well.

Of course, it is possible to use an orthogonality correction in the calculation of B_A and B_B. However, the orthogonality correction may be more economically applied to B_C instead of to B_A and B_B individually. Since this forces

$$S_s \cong S_{ss}^2/2 \ , \tag{122}$$

it follows that S_s will be compensated for, as well. If this procedure is periodically repeated, both the skew-symmetric and symmetric error will be compensated. The long-term error propagation will still be given by

$$\dot{Z}_C = R_{ss}Z_C + R_{ss} \ ; \tag{123}$$

only R_{ss} will now be an uncompensated higher order term. For a second-order algorithm, this term will be inversely proportional to f_c^4 and the algorithm will have a fourth-order error characteristic. A first-order algorithm will have a higher order symmetric term inversely proportional to f_c^3. Since this term can be eliminated by an orthonormality correction, a fourth-order error characteristic can also be achieved. The commutativity error, being inversely proportional to f_c^2, will also be compensated for.

Hopefully, the computer memory requirements are not considerably increased if the same routine is used to calculate B_A and B_B. It is not necessary to store B_C since it is an updated B_A. In addition, the time required to compute B_B will be only half that required to compute B_A because of the lower repetition rate. Since the error level is now that of a fourth-order algorithm, it is possible to reduce the basic computational frequency. For many applications the result will be an increase in accuracy, a decrease in the overall computational time, and a minor increase in memory requirements.

The compensation technique is illustrated in Figure 12. Note that the correction frequencies may be submultiples of the computation frequency. Therefore, there are many variations possible, with each yielding a different accuracy versus computer loading trade-off.

Figure 13 is the result of a simulation using the compensation technique described in this section. The simulation used a second-order Taylor series. One run was made without an orthogonality correction and one was made with an orthogonality correction applied to B_C at each step. The matrix B_B was also set equal to B_C after each correction. Figure 13 may be compared to Figures 6 and 10 since they represent the same angular rate matrix and computational frequency.

A constant rate case is shown in Figure 14. This run may be compared to Figure 3 and 7 since they all represent a first-order algorithm with the same input and computational frequency.

The first-order compensated solution of Figure 14 may also be compared with the second-order results of Figure 8. The second-order solution (using a continuous orthogonality correction) had a skew symmetric error of

$$z_{ssc} = 0.2\times10^{-1}$$

The completely compensated first order method of Figure 14 had a corresponding error of

$$z_{ssc} = 0.9\times10^{-4}$$

Table X contains a rough estimate of the relative computer time requires for both algorithms. The total time is approximately the same for both. The fully compensated algorithm, however, has

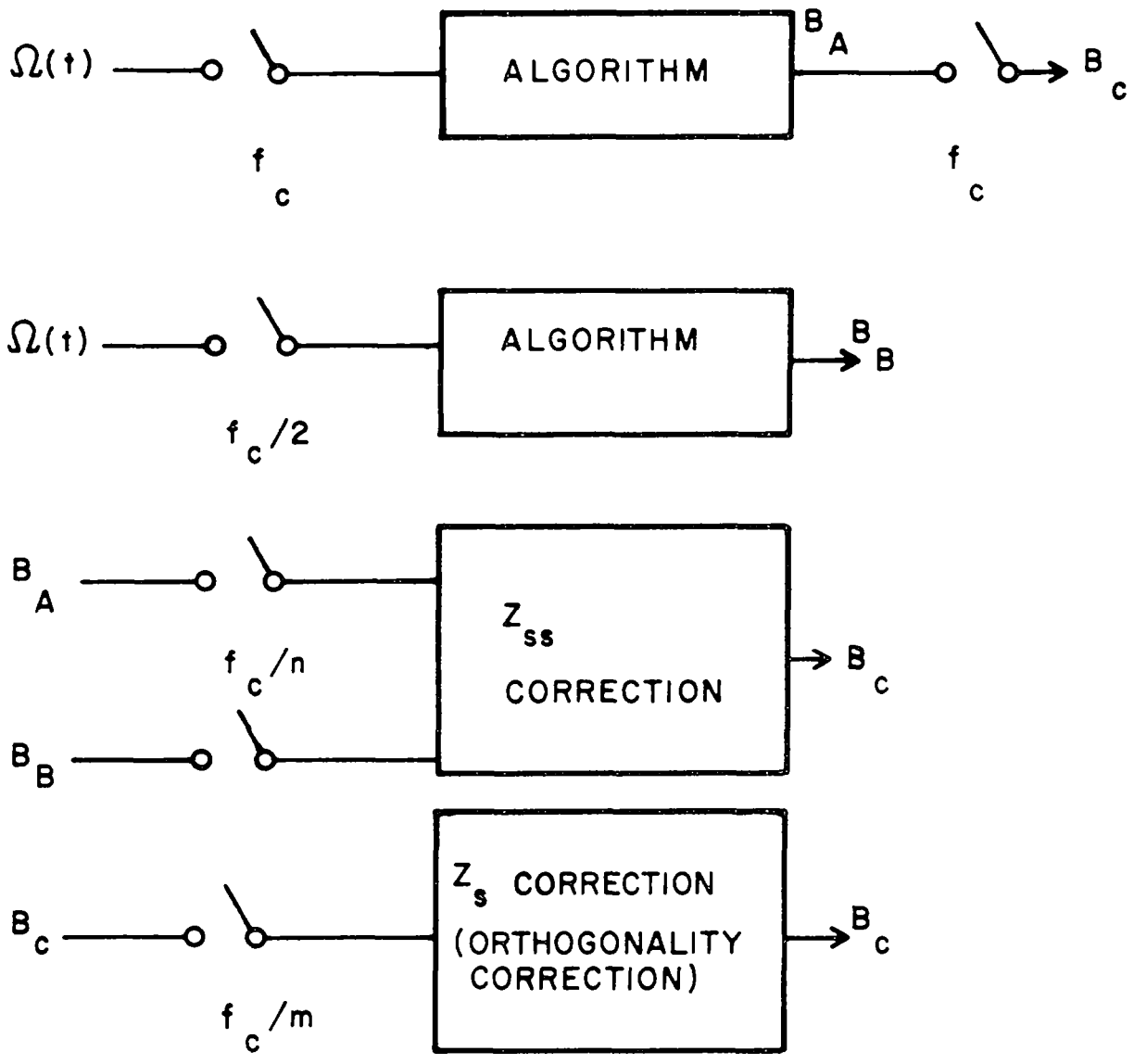


Figure 12.- Compensation technique

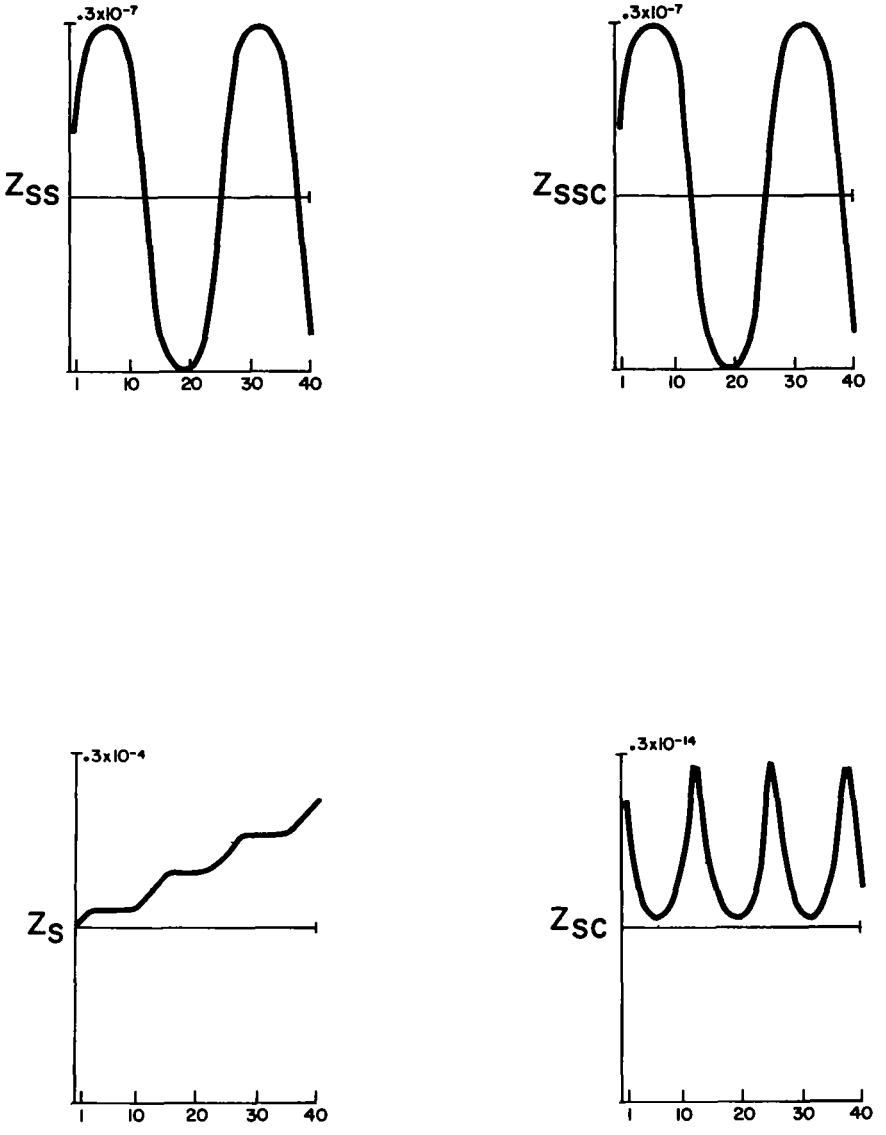


Figure 13.- Sinusoidal rates, second-order Taylor series, compensated solution

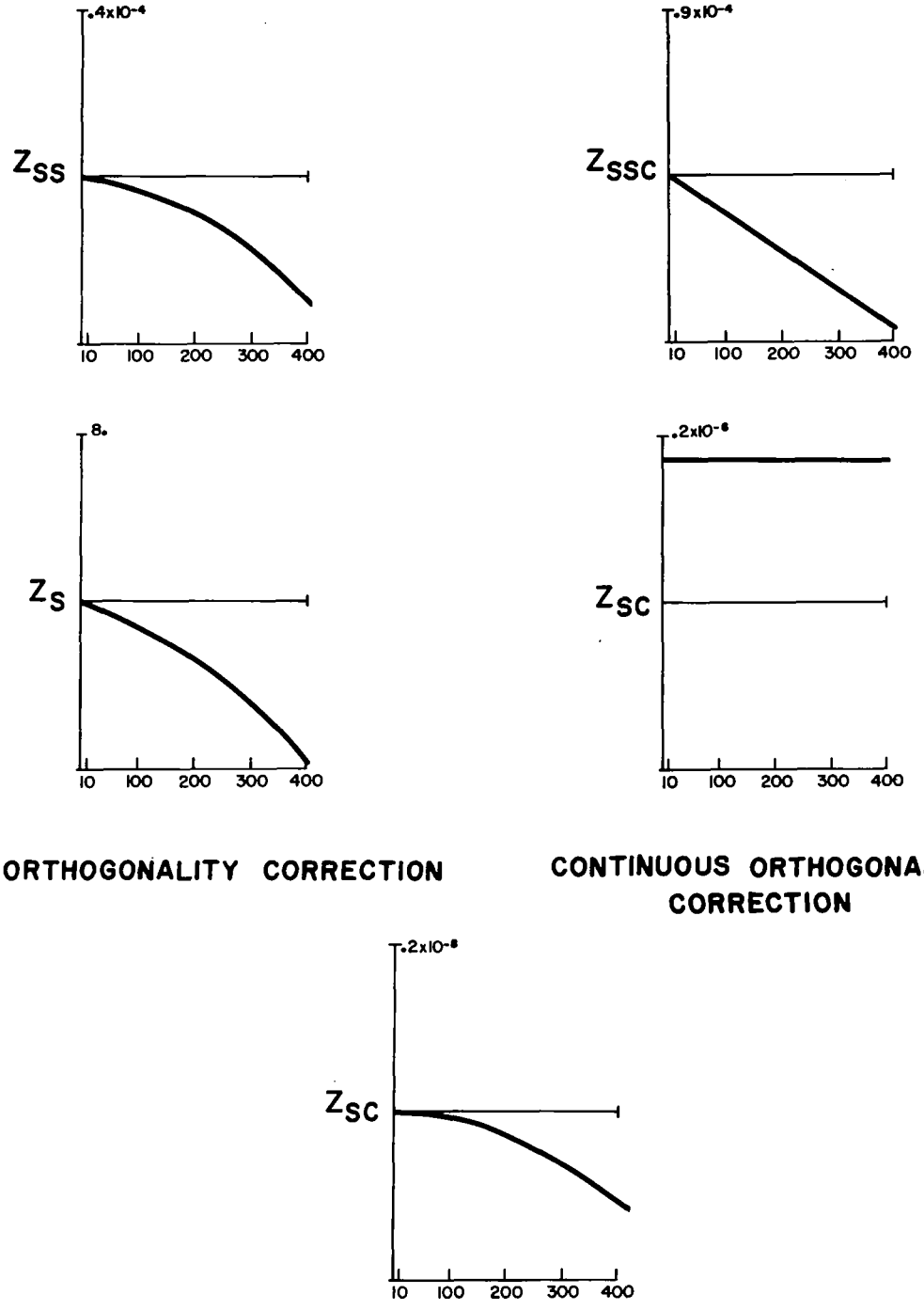


Figure 14.- Constant rates, first-order Taylor series, compensated solution

TABLE X

COMPUTATION TIMES

Calculation	2nd Order	1st (compensated)
B_A	1	0.5
B_B		0.25
B_C		0.25
Orthogonality Correction	1	1
Total	2	2

a much smaller error. If the computation frequency of the first-order algorithm were decreased by half, its error would be increased by a factor of 16 resulting in

$$z_{ssc} = 0.14\times10^{-2} \ .$$

The compensated algorithm allows a 50 percent reduction in the computational load and still retains an order-of-magnitude accuracy improvement.

Computable Solution

The fact that B is an orthogonal matrix may be used to allow the separation of the error terms in B_A and B_B.

Let

$$T2 = B_A^T B_B - I \ . \tag{124}$$

For a second-order algorithm without an orthogonality correction:

$$B_A = B(I + S_{ss} + S_s) \tag{125}$$

$$B_B = B(I + 4S_{ss} + 8S_s)$$

$$T2 = 3S_{ss} + 9S_s + (S_s - S_{ss})(8S_s + 4S_{ss})$$

or

$$T2 \cong 3S_{ss} + 9S_s \ , \tag{126}$$

provided the second order terms are small.

Then

$$S_{ss} = \frac{T2 - T2^T}{6} \tag{127}$$

$$S_s = \frac{T2 + T2^T}{18}$$

and for a first-order algorithm

$$S_{ss} = \frac{T2 - T2^T}{6} \tag{128}$$

$$S_s = \frac{T2 + T2^T}{6} \ .$$

When an orthogonality correction is used, the higher order terms cannot be neglected. The value of T2 becomes

$$T2 = 3S_{ss} + 17S_s - 4S_{ss}^2 \tag{129}$$

where the third term on the right-hand side is the most significant of the higher order terms. It is still true that

$$S_{ssc} = \frac{T2 - T2^T}{6} \quad ; \tag{130}$$

however, the calculation of the symmetric error term is more complex. For a second-order algorithm

$$S_{sc} \cong S_{ss}^2/2 \tag{131}$$

and

$$S_{sc} = \frac{T2 + T2^T}{18} \; . \tag{132}$$

Equation (132) also applies for a first-order algorithm, provided the residual error is small. If the residual error is large, then

$$S_{sc} > S_{ss}^2/2 \tag{133}$$

and

$$S_{sc} = \frac{T2 + T2^T}{34}$$

is a better estimate of the symmetric error. The factor of 18 is suitable for error analysis purposes since it will give a pessimistic estimate (by a factor of 2) for a first-order algorithm when the residual error is large. It is also possible to use

$$S_{ssc} = \frac{T2 - T2^T}{6} \tag{134}$$

$$S_{sc} = \frac{T2 + T2^T}{32} + S_{ss}^2\left(\frac{8}{32}\right)$$

which is more complex, but has the advantage of being applicable to both first- and second-order algorithms, regardless of the residual error.

Since S_{SS} and S_S approximate Z_{SS} and Z_S, these equations may be used to find a computable solution for Z_{SS} and Z_S. The solution employs a FORTRAN program. The algorithm under study is a subroutine in this program. An arbitrary rate input is integrated to obtain B_A. The matrix B_B is computed at half the computational frequency of B_A, using the same subroutine. From B_A and B_B the matrix T2 is found and then Z_{SS} and Z_S.

Unlike the case of the compensated solution (discussed in the previous section), the Z matrix is not corrected. Therefore, the S and Z matrices will differ as time passes. The inaccuracy in the computable solution can be estimated from the Numman series solution for the Z matrix

$$Z(t) = S(t) + \int_0^t R(\lambda)S(\lambda) + \ldots \qquad (135)$$

The second term can be used to estimate the inaccuracy or to apply a correction. A simpler technique for checking the validity of the computable solution is to recompute the error with a different step size and check for a proportional increase in the error.

A simulation program was written using the classical coning motion:

$$\gamma = 0.3 \qquad (136)$$

$$w_s = 0.25$$

$$w_x = w_s(1 - \cos \gamma)$$

$$w_y = w_s \sin \gamma \cos w_s t$$

$$w_z = w_s \sin \gamma \sin w_s t \ .$$

This is the only case (known to the author) where an analytic direction cosine matrix is known for a non-commutative rate matrix. A solution is contained in References 2 and 3. A summary of the work reported in reference 3 is contained in reference 4. The errors determined by simulation were compared to the errors determined by a computable solution. The results are shown in Figures 15 through 18. Since the rate matrix is not self-commutative, the error cannot be adequately described by the two scalar parameters z_{SS} and z_S. The nine separate parameters defined in the

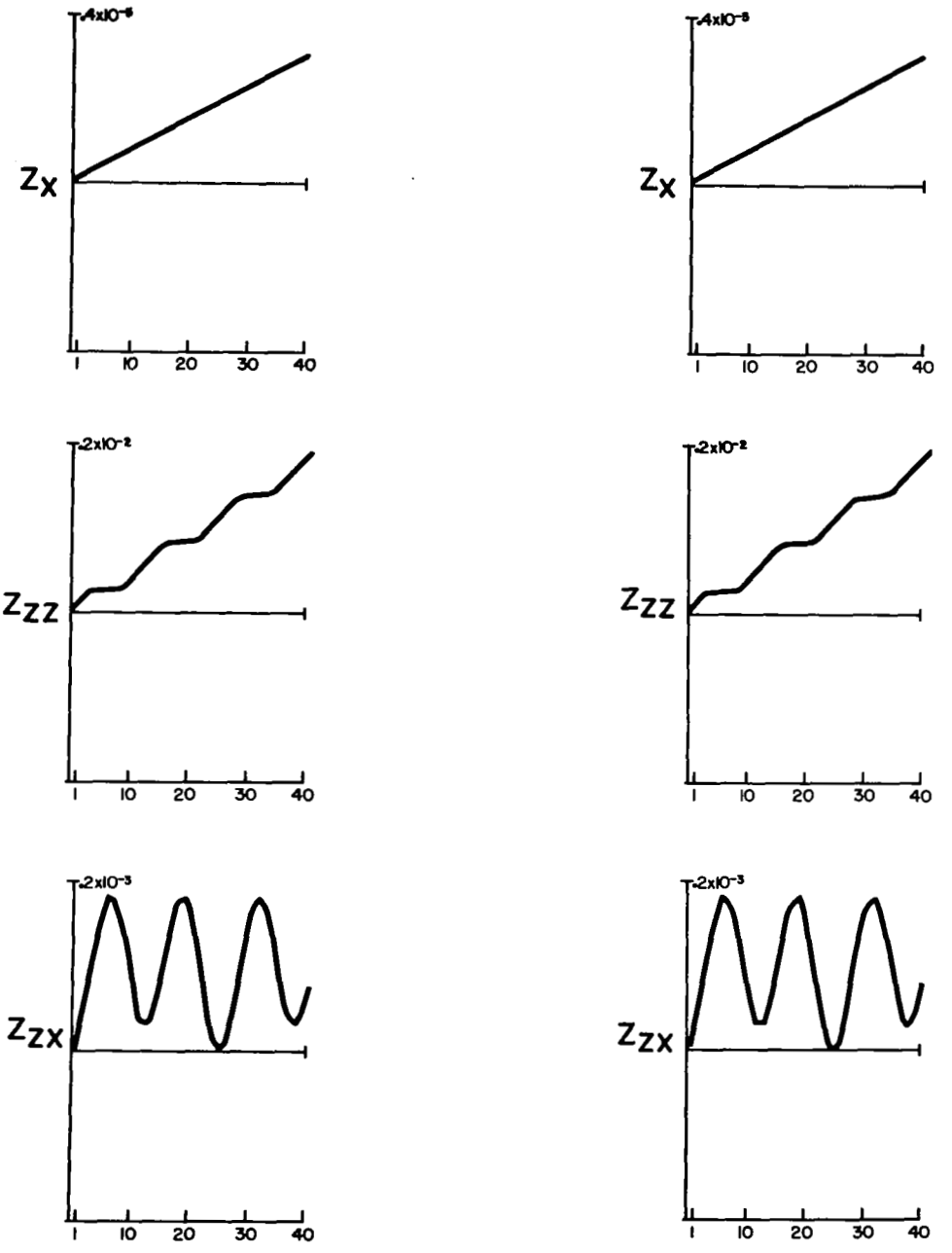


Figure 15.- Coning motion, first-order Taylor series

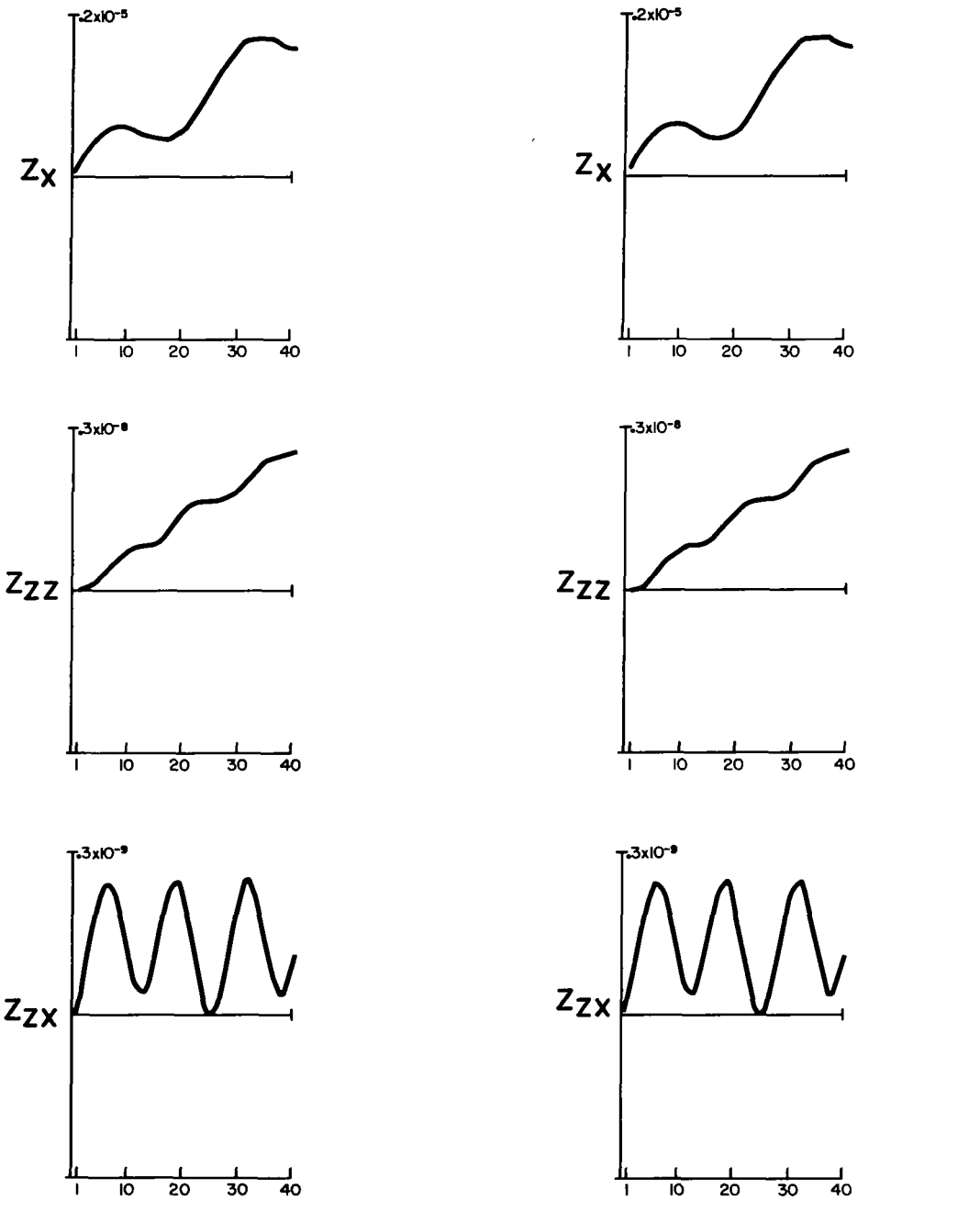


Figure 16.- Coning motion, second-order Taylor series

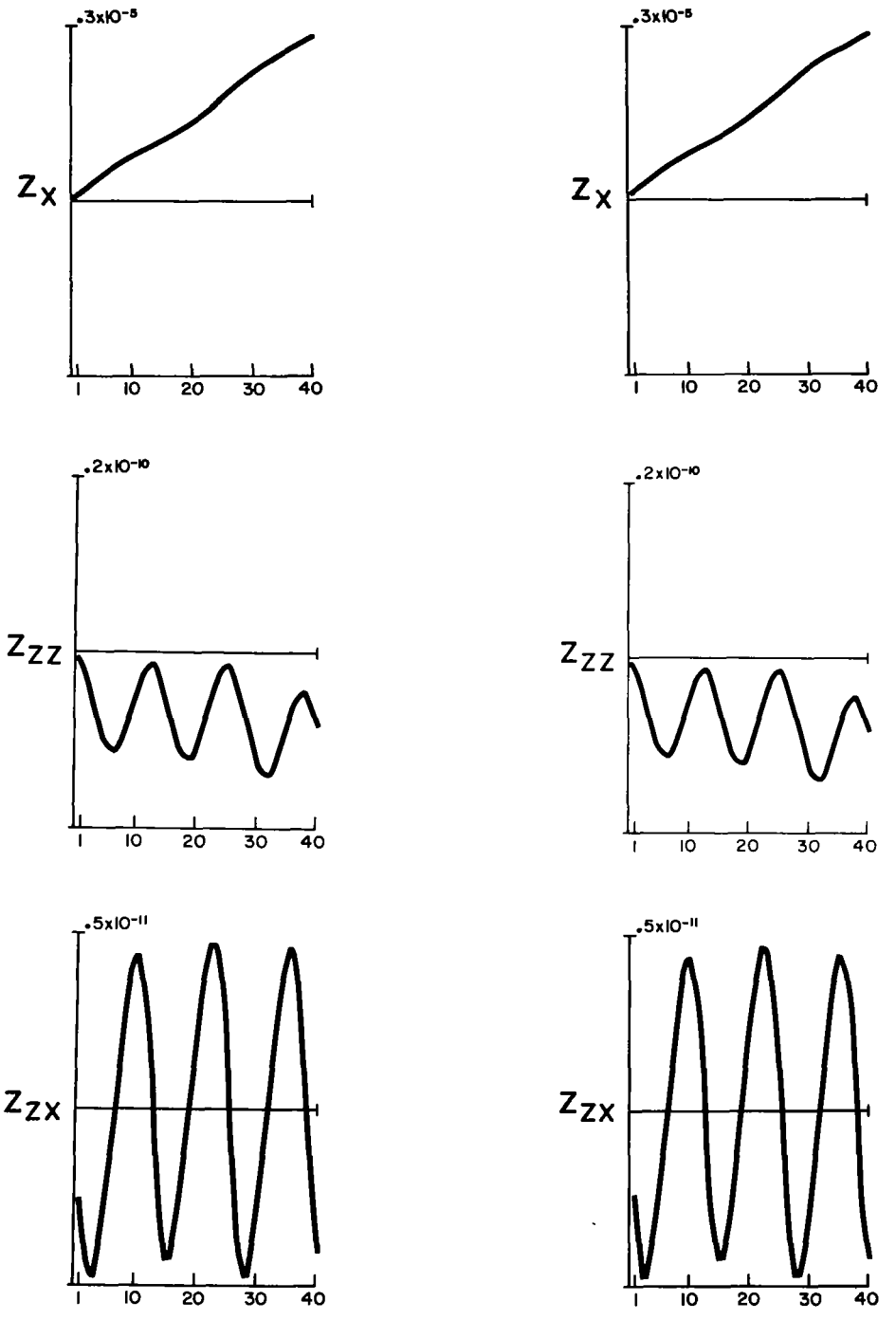


Figure 17.- Coning motion, first-order Taylor series, continuous orthogonality correction

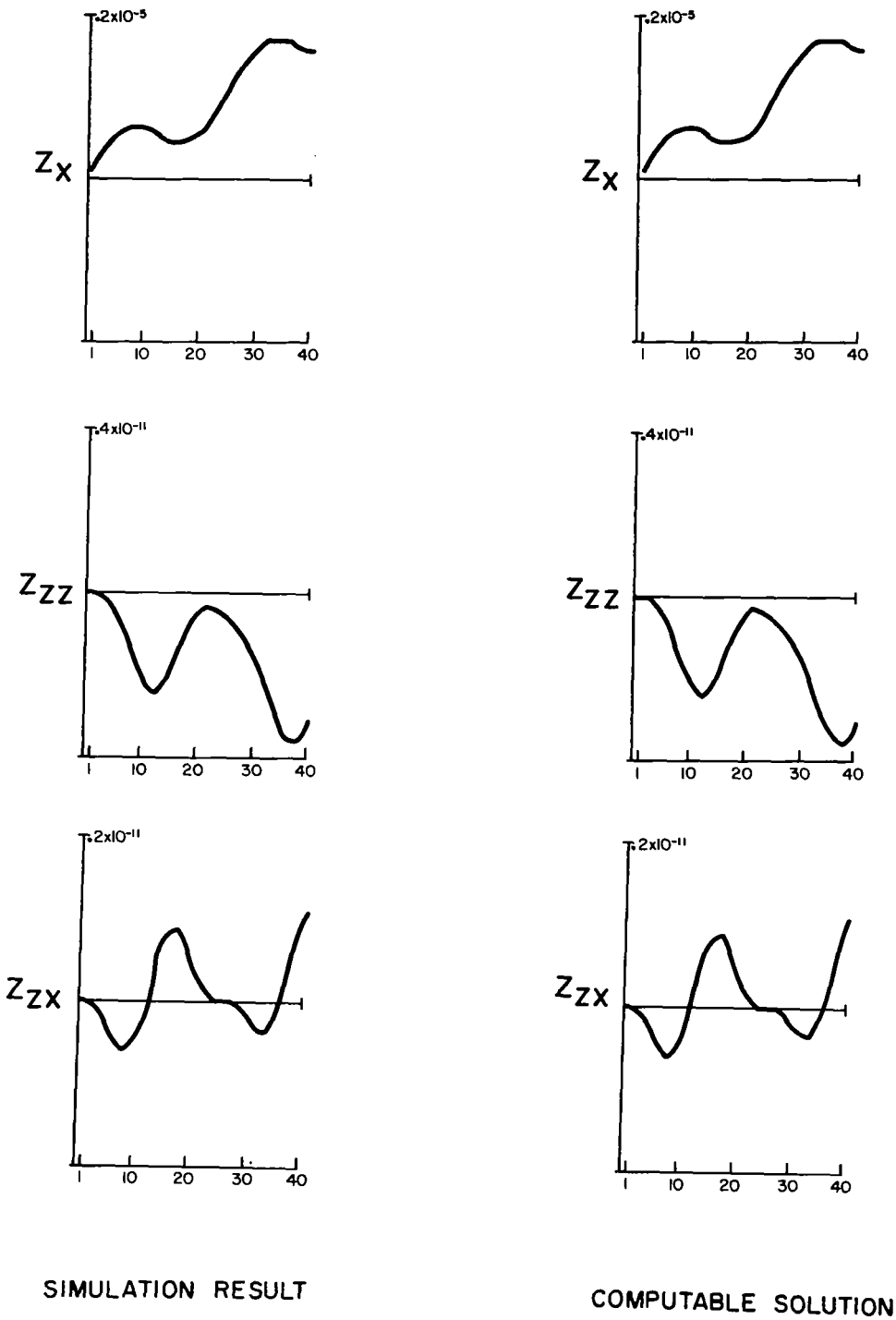


Figure 18.- Coning motion, second-order Taylor series, continuous orthogonality correction

Error Criteria section must be used. For simplicity, only three of the nine are drawn in the figures.

III. WORDLENGTH ERROR

Introduction

One of the most important factors influencing the design or evaluation of an airborne computer is an appropriate choice of wordlength. Although many factors must be considered in selecting the wordlength, the necessity of achieving adequate computational accuracy is often of decisive importance. Unfortunately, it is often quite difficult to determine the degradation of computational accuracy because of finite wordlength.

Simulation is a useful tool for this purpose. A large, general-purpose computer can be programmed to simulate the airborne computer and to execute mathematical calculations with a reduced wordlength. However, this approach is expensive, not only because of the computer running time, but more importantly, because of the programming effort involved. The basic limitation of simulation, however, is the difficulty in interpreting the simulation data without the benefit of an analytical theory. The wordlength error generated in a sequence of calculations is a function of the variables involved and will depend on the initial conditions. Changes in the initial conditions often generate very different error time histories. In these cases, it is a statistical description of the error which is of value. These statistics can be estimated by taking an ensemble average over many experiments, but, needless to say, the simulation costs may be prohibitively increased.

The development of an analytical theory likewise presents many difficulties. The wordlength error depends upon:

1. The order in which calculations are preformed,
2. The operation of the computer's arithmetic unit,
3. The fixed-point scaling used or the use of a floating point,
4. The way negative numbers are represented in the computer,
5. The initial conditions of the problem,

6. The way numbers are shortened to the required wordlength,

7. The use of double precision or extended precision operations.

Two different approaches have been taken in generating an analytical theory. One approach is exemplified by Wilkenson's book (ref. 5) on determining maximum bounds for the error. Another approach is used by Henrici (ref. 6). It deals with the statistics of the errors. In a recent paper Hull and Swenson (ref. 7) present the results of simulations designed to test the validity of the statistical model. They conclude that the models are, in general, very good. Discrepancies are both rare and mild. The discrepancies were attributed to the capability of the computer to do some operations exactly, whereas the statistical model assigns an error to all operations. Therefore, the results of the statistical model tend to be conservative. Others (refs. 2, 6, 8, 9, 10) have also reported good correlation between the theory and simulation results. In the sequel, the statistical theory of Henrici (ref. 6) will be emphasized. To simplify the discussion, the following sections will consider the propagation of wordlength error for a set of first-order differential equations. The results will then be applied to the direction cosine problem.

Propagation of Wordlength Error

Many navigational and guidance functions can be described by a set of first order differential equations

$$\dot{\underline{x}} = \underline{f}(\underline{x},t) \quad . \tag{137}$$

The differential equations will be assumed to be linear and of the form

$$\dot{\underline{x}}(t) = F(t)\underline{x}(t) \tag{138}$$

where F(t) is, in general, a time-varying matrix. For convenience in digital computation, the differential equations can be converted into linear difference equations of the form

$$\underline{x}_k = \Phi_k \underline{x}_{k-1} \tag{139}$$

$\underline{x}_k$ is the n dimensional state vector, and

Φ_k is the n×n state transition matrix.

The digital computation will not, however, generate the true state vector $\underline{x}_k$ because of the error introduced by the finite computer wordlength. Instead, a computed state vector $\underline{x}_k^*$ will result

$$\underline{x}_k^* = \Phi_k \underline{x}_{k-1}^* + \underline{\varepsilon}_k \qquad \underline{x}_o^* = 0 \tag{140}$$

where

$\underline{x}_k^*$ is the computed state

$\underline{\varepsilon}_k$ is the n dimensional wordlength error introduced at the kth stage

Now, let

$$\underline{e}_k = \underline{x}_k^* - \underline{x}_k \tag{141}$$

be the state error vector. Subtracting Eq. (139) from (140) gives:

$$\underline{e}_k = \Phi_k \underline{e}_{k-1} + \underline{\varepsilon}_k \tag{142}$$

for the propagation of the state error. The statistical theory of wordlength error assumes that $\underline{\varepsilon}_k$ is a random vector, the value of which at any time is unknown. The mean value and variance of the error are assumed to be known. The mean value of the state error is

$$\bar{\underline{e}}_k = \Phi_k \bar{\underline{e}}_{k-1} + \bar{\underline{\varepsilon}}_k \tag{143}$$

Let:

$$\underline{m}_k = -\bar{\underline{e}}_k \tag{143}$$

$$\underline{\alpha}_k = -\bar{\underline{\varepsilon}}_k \ .$$

Then:

$$\underline{m}_k = \Phi_k \underline{m}_{k-1} + \underline{\alpha}_k \qquad m_o = 0 \tag{145}$$

Eq. (145) is the basic equation for the propagation of the mean value of the state error. To determine variations about the mean value, let

$$\underline{\eta}_k = \underline{e}_k - \bar{\underline{e}}_k \tag{146}$$

$$\underline{q}_k = \underline{\varepsilon}_k - \bar{\underline{\varepsilon}}_k \quad .$$

It follows from Eq. (143) that

$$\underline{\eta}_k = \Phi_k \underline{\eta}_{k-1} + \underline{q}_k \ , \tag{147}$$

and the covariance matrix of the variation about the mean value is

$$P_k = \overline{\underline{\eta}_k \underline{\eta}_k^T} \tag{148}$$

$$P_k = \Phi_k P_{k-1} \Phi_k^T + \overline{q_k q_k^T} + \Phi_k \overline{\eta_{k-1} q_k^T} + \overline{q_k \eta_k^T} \Phi_k^T \ .$$

Writing

$$Q = \overline{q_k q_k^T}$$

and making the assumption that

$$\Phi_k \overline{\eta_{k-1} q_k^T} + \overline{q_k \eta_{k-1}} \Phi_k^T = 0 \ , \tag{149}$$

the covariance matrix becomes

$$P_k = \Phi_k P_{k-1} \Phi_k^T + Q_k \ ; \qquad P_o = 0 \tag{150}$$

Equations (145) and (150) are two key relations since they allow a recursive calculation of the mean and covariance of the state error resulting from the wordlength error. They involve only the statistics of the error and do not require simulations to determine the actual errors.

Continuous Model

Although the computer calculates in a discrete fashion, it is also possible to propagate the wordlength error by a continuous model (ref. 6). In this case the basic difference equation

$$\underline{x}_k = \Phi_k \underline{x}_{k-1} \tag{151}$$

is replaced by the differential equation

$$\dot{\underline{x}}(t) = F(t)\underline{x}(t) \tag{152}$$

where F(t) is in general a time-varying matrix. Since the relationship between the discrete and continuous model is well known (refs. 10,11,12) the key equations are listed in Table XI without derivation.

TABLE XI

KEY EQUATIONS

Discrete Model:
$\underline{m}_k = \Phi_k \underline{m}_{k-1} + \underline{\alpha}_k \qquad m_o = 0$
$P_k = \Phi_k P_{k-1} \Phi_k^T + Q_k \qquad P_o = 0$
Continuous Model:
$\dot{\underline{m}}(t) = F(t)\underline{m}(t) + f_c \underline{\alpha}(t) \qquad m(0) = 0$
$\dot{P}(t) = F(t)P(t) + P(t)F^T(t) + f_c^2 Q(t) \qquad P(0) = 0$

In the continuous model of the mean value equation, the mean value of the wordlength error is given by $f_c\alpha(t)$. The parameter f_c is the computational frequency with which the discrete equations are being solved by the computer, i.e., the number of wordlength errors generated per unit time.

Both the continuous and discrete models provide a computable solution for the error statistics rather than a closed form solution. For some applications, the continuous model can be used to find closed-form solutions. Table XII contains the symbolic solution for the continuous model.

TABLE XII

SOLUTION FOR THE CONTINUOUS MODEL

$$\dot{\Phi}(t,t_0) = F(t)\Phi(t,t_0) \qquad \Phi(t_0,t_0) = I$$

$$\underline{m}(0) = 0$$

$$\underline{m}(t) = \Phi(t,t_0)\underline{m}(t_0) + f_c\Phi(t,t_0)\int_{t_0}^{t} \Phi^{-1}(\tau,t_0)\underline{\alpha}(\tau)\, d\tau$$

$$P(0) = 0$$

$$P(t) = \Phi(t,t_0)P(t_0)\Phi^T(t,t_0) + f_c^2\int_{t_0}^{t} \Phi(\tau,t_0)Q(\tau)\Phi^T(\tau,t_0)\, d\tau$$

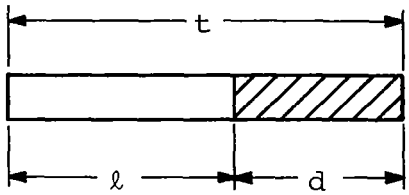
The results contained in Tables XI and XII are essentially those given by Henrici in reference 5. To use either the continuous or discrete model, it is necessary to obtain expressions for the mean value and covariance of the word length error vector. In the next section the theory contained in reference 6 will be extended to include different computer number systems (i.e., complement representations as well as sign-magnitude) and different organizations of the computer's arithmetic unit. The extension will first be applied to fixed-point computers. A separate section will then extend the results to floating-point machines.

Fixed Point Computer

Origin of wordlength error.- Wordlength error may result whenever the number of bits representing a number is reduced. Each computer operation must be examined individually to determine whether or not it involves a reduction in significant bits. First, however, the process of shortening a single word will be analyzed.

Positive numbers.- Consider a positive fixed point number x which is t bits in length. It is to be shortened to ℓ bits. The process of shortening may be done in two different ways, each of which generates errors in different ranges.

Chopping.- The t bit word is simply chopped off to ℓ bits.



In the chopping operation, d bits are lost which have some positive value r. If x lies in the range $0 < x < 1$, then the maximum value of the error introduced by chopping will be

$$\text{Max } r = -2^{-\ell} .$$

In general, x will be scaled at some value 2^v, so that the maximum value of the error is actually

$$\text{Max } r = -2^{v-\ell}$$

The minimum value of the error "zero" occurs when the lost bits have the value "0". The actual value of the error lies somewhere between these two limiting values. The statistical theory of wordlength error assumes that the error is uniformly distributed between the extremes. Letting $k = 2^{v-\ell}$, the probability distribution of the error is shown in Figure 19. For positive x the wordlength error ε is simply the negative of the value of the lost bits or as shown in Figure 19.

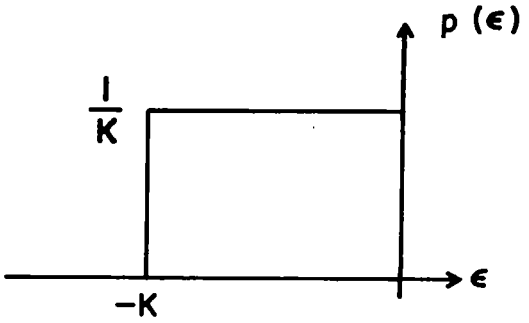


Figure 19.- Distribution of wordlength error when a chopping operation is performed on a positive number.

The error distribution shown in Figure 19 has an average value of

$$\bar{\varepsilon} = -\frac{k}{2} .$$

This characteristic of the error caused by chopping is often highly undesirable. The second method of shortening wordlength does not have this drawback.

Symmetrical rounding.- Before the t bit word is shortened to ℓ bits, the value k/2 is added. The wordlength is then shortened to ℓ bits.

The rounding operation shifts the distribution of the wordlength error to the right so that it has zero mean value.

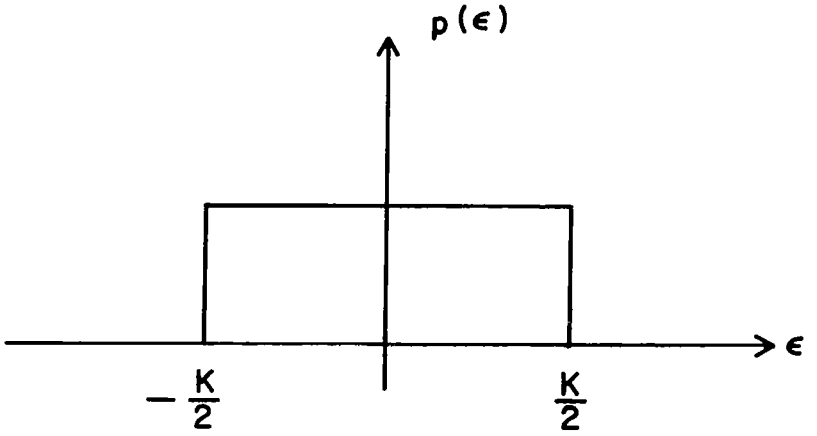


Figure 20.- Distribution of wordlength error when a symmetrical rounding operation is used.

Negative numbers.- For the case where x is a negative number the results are slightly more complex. The sign of the wordlength error will depend on the method of representing negative numbers in the computer, i.e., sign and magnitude, one's complement or two's complement representations. (Positive numbers have identical representations for all numbers systems.)

Sign and magnitude.- In this system, a negative number is represented by its absolute value plus a sign bit. A chopping operation will always reduce the absolute value of the number. Therefore, if the original number x is negative, the chopping operation will result in a less negative number or a positive error. Thus, the error for both positive and negative x can be written as

$$\varepsilon = -r \, \mathrm{sgn}(x) \tag{153}$$

where

$$\mathrm{sgn}(x) = 0 \qquad x = 0$$
$$\mathrm{sgn}(x) = -1 \qquad x < 0$$
$$\mathrm{sgn}(x) = 1 \qquad x > 0 \; .$$

For negative values of x the probability distribution of the error shown in Figure 19 is reflected about the vertical axis since the error will always have a sign opposite to the sign of x. A one's complement machine will have the same error characteristics as a sign and magnitude machine.

Two's complement.- According to reference 13, the signed two's complement representation of a negative binary number will give the true value of the binary number if the value of the sign bit is regarded as negative. For example, the number 1,1001 has the value

$$-10000 + 1001 = -0111$$

A wordlength error will result in a more negative result giving a negative error. Therefore, when negative numbers are represented in two's complement form, the wordlength error caused by chopping will always appear as in Figure 19 and will be independent of the sign of x.

Symmetrical rounding.- Finally, since symmetrical rounding produces an error distribution which is symmetrical about zero, the sign of x is of no consequence irrespective of the number system employed.

Statistics of the wordlength error.- The variance of a variable uniformally distributed between 0 and k is given by

$$\overline{\varepsilon^2} = \frac{k^2}{12} \quad .$$

The mean value and variance of the wordlength error are summarized in Table XIII. Use is made of the definitions

$$\alpha = -\overline{\varepsilon} \quad \text{and} \quad q = \overline{\varepsilon^2}$$

TABLE XIII

ERROR STATISTICS

Number System	Chopping		Symmetric Rounding	
	α	q	α	q
S/M	$\frac{k}{2}\ \mathrm{sgn}(x)$	$\frac{k^2}{12}$	0	$\frac{k^2}{12}$
2C	$\frac{k}{2}$			

Computer mechanization.- The errors introduced by the fundamental arithmetic operations may be analyzed in terms of the shortening of wordlength discussed in the previous sections.

Although the system is described by the difference equation

$$\underline{x}_k = \Phi_k \underline{x}_{k-1} \quad , \tag{154}$$

this is not the usual form of implementation on a digital computer. A more common procedure involves the numerical integration

of the basic differential equations. This process may be written as

$$\underline{x}_k = \underline{x}_{k-1} + \Delta\underline{x}_k \tag{155}$$

where the increment Δx is determined by the numerical integration formula. To preserve accuracy, the step size is chosen small enough so that Δx is a relatively small quantity. Five different methods of performing the addition are of interest:

SPSA - Single Precision Short Accumulator

SPLA - Single Precision Long Accumulator

DPSA - Double Precision Short Accumulator

DPLA - Double Precision Long Accumulator

EP - Extended Precision

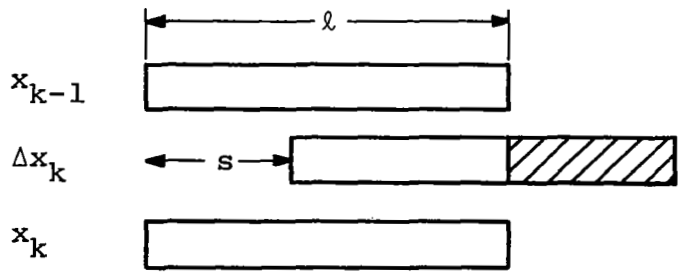
Single precision short accumulator.- Since Δx is small relative to x, assume that it is calculated with a smaller scale factor:

2^v scale factor of x

2^γ scale factor of Δx

$s = v - \gamma$.

Then Δx must be shifted s bits to the right before it is added to x_{k-1}



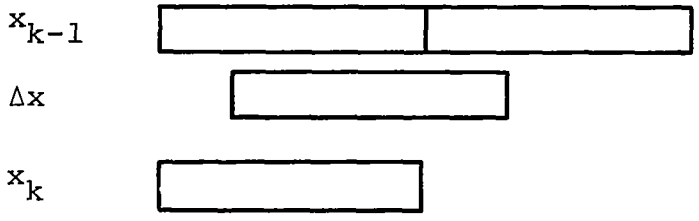
The number Δx_k is shortened to ℓ-s bits. For sign and magnitude representation, the maximum error would be

$$\max \varepsilon = -2^{\gamma-\ell-s} \operatorname{sgn}(\Delta x) = -2^{v-\ell} \operatorname{sgn}(\Delta x).$$

The maximum value of the error depends not on the scaling of Δx, but rather on the scaling of x. The sign is opposite that of Δx.

It has been assumed that the computation of Δx was carried out with an optimum scale factor. Wordlength error associated with the computation of Δx would then affect only the least significant bits of Δx and would be "shifted off" when the addition to x is performed. It is possible to program so that an explicit shifting of Δx is not required. In this case, the shift operation is performed by a multiplication during the calculation of Δx.

Single precision long accumulator.- Assume that all variables are of single precision with a length of ℓ bits. However, the accumulator into which x_{k-1} is loaded is extended with additional bits which are equal to zero

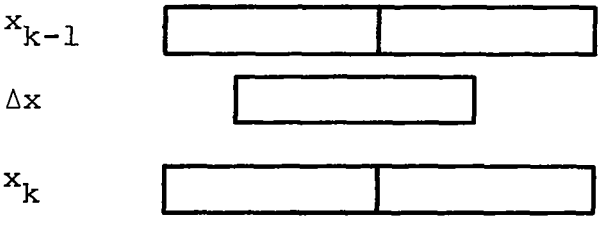


The final result x_k is shortened by chopping or symmetrical rounding. For sign and magnitude arithmetic the maximum error would be

$$\max \varepsilon = -2^{v-\ell} \operatorname{sgn}(x) \quad .$$

The magnitude of the error is the same as for single precision addition, but the sign is now determined by x rather than Δx. For a two's complement number system, or if symmetrical rounding is used, long accumulator addition does not differ from short accumulator addition.

Extended precision addition.- For extended precision addition, the variable x is stored in the computer by two ℓ bit words. Even if only the most significant ℓ bits are used in other calculations, the double length word is available when Δx is added to x_{k-1}:



The wordlength error results from shortening Δx and for a sign and magnitude number system is

$$\max \varepsilon = -e_{\Delta x} \operatorname{sgn}(\Delta x)$$

where $e_{\Delta x}$ is the wordlength error generated in calculating Δx. It must be estimated for each separate application. The magnitude of the error depends on the scaling of Δx as does its sign.

It is not necessary to think of extended precision operations as requiring a double length word to represent x. One may consider x as a single length word. Extended precision addition then requires a separate memory location to accumulate the lower bits of Δx which would be "shifted off" in doing a single-precision addition to x. It is then a matter of personal preference whether or not the memory location which accumulates the lower order bits of Δx are considered to be a double length extension of the variable x.

Double precision addition.- If all the variables are represented in double precision form (2ℓ bits) the word length error will be the same as for single precision addition with ℓ replaced by 2ℓ.

Summary.- The error caused by the different forms of addition are summarized in Table XIV. For a two's complement number system, the errors differ, at most, by a constant. In the table, it has been assumed that the error in calculating Δx is bounded by

$$e_{\Delta x} \leq 2^{w} .$$

The value of w is highly program-dependent and must be estimated for each individual application.

Floating Point Computers

In a floating point computer, the scaling of the variables is itself a variable. The scaling of the variable x is given by

$$2^{v-1} < x < 2^{v}$$

where v is a computed quantity. The error statistics of Tables XIII and XIV are valid for floating-point as well as fixed-point

TABLE XIV

FIXED-POINT ADDITION

SPSA/DPSA		
Number System	SM	Two's Complement
k	$2^{v-\ell}/2^{v-2\ell}$	$2^{v-\ell}/2^{v-2\ell}$
α(chop)	$k/2\ \mathrm{sgn}(\Delta x)$	$k/2$
α(round)	0	0
q^2	$k^2/12$	$k^2/12$

SPLA/DPLA		
Number System	SM	Two's Complement
k	$2^{v-\ell}/2^{v-2\ell}$	$2^{v-\ell}/2^{v-2\ell}$
α(chop)	$k/2\ \mathrm{sgn}(x)$	$k/2$
α(round)	0	0
q^2	$k^2/12$	$k^2/12$

EP		
Number System	SM	Two's Complement
k	$2^{w-\ell}$	$2^{w-\ell}$
α(chop)	$k/2\ \mathrm{sgn}(\ x)$	$k/2$
α(round)	0	0
q^2	$k^2/12$	$k^2/12$

computations with the exception that v and w must be regarded as variables for the floating point calculations. If the equations of Table XI for the propagation of the wordlength error are solved on a computer, the fact that v is a variable represents only a slight complication. For example, on a digital computer, it is only necessary to include a subroutine which accepts a state variable as its input and determines 2^v for that variable.

Closed-Form Solutions

Covariance of the error.- Closed-form solutions can often be found by performing the integrations indicated in Table XII. The solution of the covariance equation is the same for all number systems. It is the complete solution if symmetric rounding is used since the mean value of the error will be zero. A simplification results if the initial reference time t_o is equal to zero, since the first term in the formula for the covariance vanishes.

Mean value of the error.- The mean value need only be computed when a chopping operation is used, since it is zero when a rounding operation is employed. Complications are introduced for a sign and magnitude number system since the sign of the error will be a variable. The integration must be divided into time periods which have the same error sign and a piecewise integration carried out.

Computable Solutions

The value of closed-form solutions lies in the insight they provide into the propagation of the wordlength error. For all but the simplest systems, however, the closed-form solutions can involve tedious algebraic manipulations. These can be avoided if a computable rather than a closed-form solution is employed for analysis. The computable solution may be solved on a general-purpose digital computer or on an analog computer.

Discrete computable solution.- A generalized computer program may be written which propagates the difference equations:

$$\underline{x}_k = \Phi_k \underline{x}_{k-1} \tag{156}$$

$$\underline{m}_k = \Phi_k \underline{m}_{k-1} + \underline{\alpha}_k \qquad m_0 = 0$$

$$P_k = \Phi_k P_{k-1} \Phi_k^T + Q_k \qquad P_0 = 0 \quad .$$

The initial state x_0 and the state transition matrix are inputs to the program. If Φ varies with time, then a subroutine may be added to calculate the state transition matrix. The quantities $\underline{\alpha}_k$ and Q_k are calculated in accordance with Table XIV. Some of the quantities in Table XIII depend on the current state $\underline{x}_k$. For example, if sign and magnitude numbers are assumed, the wordlength error will depend on sgn($\underline{\Delta x}$). In this case, the subroutine calculating $\underline{\alpha}_k$ will use two values of the state vector to calculate:

$$\underline{\Delta x}_k = \underline{x}_k - \underline{x}_{k-1} \cdot \tag{157}$$

The reader is cautioned that the function sgn(x) is defined with

$$\text{sgn}(0) = 0$$

The sign function found in most FORTRAN system is similar to sgn but it may have the value ±1 when its argument is zero. It will then be necessary to test for a zero argument before using the sign function.

When a floating-point number system is being analyzed, the value of k in Table XIII will be a variable. This presents no problem for the computable solution. The current value of k may be found from the current state by a masking operation. A mask is applied to the variable $\underline{x}_k$ so that its exponent is left unchanged, while its mantissa becomes the number +1. The masking is done by a FORTRAN logical operation and no assembly language programming is necessary. The output of the program is a plot of $\underline{x}_k$, $\underline{m}_k$, and P. The value of P may be interpreted as the variance of the error which would result if a rounding operation were used. If chopping is used, the error is given by the plot of the mean value of the state error ($\underline{m}$) while P represents the variance of variations about the mean.

Continuous computable solution.- A continuous solution may be found by solving the equations

$$\dot{\underline{x}}(t) = F(t)\underline{x}(t) \tag{158}$$

$$\dot{\underline{m}}(t) = F(t)\underline{m}(t) + f_c\underline{\alpha}(t)$$

$$\dot{P}(t) = F(t)P(t) + P(t)F^T(t) + f_c^2Q(t)$$

on a digital or analog computer. This form has the advantage that the state transition matrix need not be known. It allows a solution for cases where no analytical state transition matrix can be found. In addition, the continuous model may be integrated on a digital computer with a variable step size algorithm.

Summary.- Either the discrete or continuous computable solutions offer a very general and simple method for determining the propagation of the wordlength error. Table XII shows that the mean value and standard deviation are both linear functions of f_c and $2^{-\ell}$. Therefore, the dependence on the parameters f_c and ℓ is known. In addition, it is usually possible to obtain some analytical results for at least a simplified model of the system of interest. These factors usually ensure that a high degree of confidence can be placed in the computable solution.

Direction Cosine Wordlength Error

The propagation of the direction cosine matrix is given by

$$\dot{B} = \Omega B \ .$$

The fact that a digital computer has a finite wordlength will result in the calculation of an inaccurate cosine matrix denoted by $\hat{B}$. Although the error is random in nature, it is possible to calculate the mean value and variance of the error.

Error propagation with symmetric rounding.- If the computer employs a symmetric rounding operation, the mean value of the state error vector will be zero and need not be calculated. The actual error will have a random nature which is characterized by the covariance matrix P(t) of the state error vector. Since the B matrix consists of direction cosines, the maximum value of any element is one. (Actually, the variables must be scaled slightly larger to allow for computational error; however, unity scaling will suffice for an error analysis.)

Let the state variables be the first column of the B matrix and assume that a fixed-point computer is being used. If the wordlength is ℓ bits, then from Table XIV

$$k = 2^{-\ell} \ .$$

Since all state variables have the same scaling, the Q matrix

(Table XIII) is:

$$Q = \frac{k^2}{12} I \tag{159}$$

where I is the unit matrix. This value applies to single precision and double precision (with ℓ replaced by 2ℓ). With extended precision addition the magnitude of the error is proportional to the scaling of the state variable increment instead of the scaling of the state variable. The value of k may then differ by a power of 2. The general form of the solution will be the same. A closed-form solution for the covariance matrix can be found from Table XII.

$$P(0) = 0 \tag{160}$$

$$P(t) = f_c^2 \int_0^t B(x)Q(x)B^T(x)\, dx$$

$$P(t) = \frac{f_c^2 k^2}{12} \int_0^t B(x)B^T(x)\, dx ,$$

and, since B is an orthogonal matrix:

$$P(t) = \frac{f_c^2 k^2}{12} tI . \tag{161}$$

The covariance matrix of the error vector, because of finite wordlength, remains diagonal and grows linearly with time. The errors in the state variables remain uncorrelated. A similar analysis can be made for the two remaining columns of the B matrix. Because of the identical scaling, the three columns of the error matrix [the E matrix of Eq. (6)] will be identical. Since the error is a statistical quantity, it is necessary to generalize the definition of the error matrix.

Defining:

$$U = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} , \tag{162}$$

the E matrix may be defined as

$$E = \sigma U$$

where

$$\sigma = \frac{f_c^k}{\sqrt{12}} \sqrt{t} \tag{163}$$

is the standard deviation (one sigma value) of the error. (Two or three sigma values can also be used.) Because the errors are uncorrelated, the Z matrix is also

$$Z = \sigma U \ .$$

When symmetric rounding is used, the resultant errors are uncorrelated with a standard deviation that grows with the square root of time.

Mean value of the error.- If the computer wordlength is shortened by a chopping operation, the mean value of the error will, in general, be non-zero. The propagation of the estimated direction cosine matrix is given by

$$\dot{\hat{B}} = \Omega \hat{B} + A \tag{164}$$

where A is a matrix containing the average value of the wordlength errors introduced in one computational step. From Table XI, it follows that

$$\dot{E} = \Omega E + A \tag{165}$$

and

$$\dot{Z} = B^T A$$

describe the propagation of the mean value of the direction cosine errors. Define the sgn function of a matrix as

$$\text{sgn}\ (\Phi) = [\text{sgn}\ (\phi_{ij})] \tag{166}$$

so that sgn (Φ) is a matrix derived from the matrix Φ by replacing each element of Φ by the values -1, 0, +1, depending on whether the element is less than, equal to, or greater than zero, respectively. Then Table XV contains the A matrix for several different number systems. The value of k is given for a fixed-point computer. When a floating point computer is used, k becomes a variable.

TABLE XV

WORDLENGTH ERROR MATRIX

Number System	S/M SPSA	S/M SPLA	2's Comp	$U = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$
A	$\frac{k}{2}\ \text{sgn}(\Phi)$	$\frac{k}{2}\ \text{sgn}(\Delta\Phi)$	$\frac{k}{2}U$	$k = 2^{-\ell}$

Closed-form solutions.- Closed-form or approximate solutions for the mean value of the wordlength error are possible. For example, the mean value of the two's complement wordlength error is

$$Z^T = \frac{k}{2} U \int_0^t B(x)\ dx\ . \tag{167}$$

For a constant rate about each axis:

$$\Omega = K \tag{168}$$

$$B = I + \frac{\sin w_0 t}{w_0} K + \frac{(1 - \cos w_0 t)}{w_0^2} K^2$$

and

$$Z^T = \frac{k}{2} U \left[It - \frac{(\cos w_0 t - 1)}{w_0^2} K + \frac{(w_0 t - \sin w_0 t)}{w_0^3} K^2 \right]$$

A direct evaluation shows that the skew-symmetric component of the Z matrix will remain equal to zero if the rates about all three axes have the same amplitude. Without the analytic solution, the results of a simulation run or the computable solution with equal rate inputs might be misleading. The two's complement wordlength error will generally have both a symmetric and a skew-symmetric component.

For a sign and magnitude number system, exact analytical solutions are quite difficult to obtain because of the sgn function. However, approximate solutions are easily obtained.

S/M - SPSA

Let:

$$\text{sgn } (\Delta\Phi) \cong \dot{B} = B \ .$$

Then:

$$\dot{Z} = \frac{k}{2} B^T \Omega B \ ,$$

and if Ω and B commute

$$Z = \frac{k}{2} \int_0^t \Omega(x) \, dx \ .$$

For constant angular rates:

$$Z = \frac{k}{2} \Omega t \ .$$

S/M - SPLA

Let:

$$\text{sgn } (\Phi) \cong B \ .$$

Then:

$$\dot{Z} = \frac{k}{2} B^T B$$

and

$$Z = \frac{k}{2} It$$

for all angular rate inputs.

Sign and magnitude arithmetic with a long accumulator has desirable error properties since the error propagation is symmetric and therefore subject to compensation by an orthogonality correction. Short accumulator addition, on the other hand, results primarily in a skew-symmetric error which is undesirable. In both cases, there will be second-order terms not included in the approximate solution.

Computable solution.- A computable solution for the direction cosine error may be achieved by integrating the equation set

$$\dot{B} = \Omega B \qquad (169)$$

$$E = \Omega E + A$$

or the equation set

$$\dot{B} = \Omega B$$

$$\dot{Z} = B^T A \ .$$

The details of a computer program, which will carry out this integration, have been explained previously.

Simulation results.- The validity of the statistical model of wordlength error has been upheld by the reports contained in references 6 through 10. However, it is wise to do at least a limited amount of simulation for a general class of applications. A simple validation procedure was applied to the direction cosine problem. A FORTRAN simulation program which numerically integrated the direction cosines using a second-order Taylor's algorithm was written. The simulation was done in both single precision and double precision. The single and double precision results were differenced to determine the single precision wordlength error. The program was run on a one's complement machine (Univac 1108) and on a sign and magnitude machine (IBM 7094). The results were than compared to the error which was predicted by a computable solution. Figures 21 and 22 contain plots of three elements of the Z matrix obtained from the simulation and computed solution. The sign and magnitude computer has a long accumulator floating point unit (although it uses a short accumulator mechanization for fixed point addition) and therefore exhibits predominantly symmetric error terms in the Z matrix.

The one's complement machine has a predominantly skew-symmetric error characteristic of short accumulator addition. Both machines also have smaller errors of the opposite type. The secondary errors are less accurately estimated by the computable solution. This is because the computable solution models only the principal source of the wordlength error – the addition of the state variable increment to the state variable. If this error is reduced by double precision or extended precision addition, or if a more accurate evaluation of the secondary errors is desired, the errors in the increment must also be included. This may be done by completely analogous techniques. The terminal values of the plots are given in Table XVI. The asterisks mark the errors which were predicted by the approximate analytical solutions.

The simulations were also run with a continuous orthogonality correction. The results are given in Figure 23. They illustrate the suppression of the symmetric error component by the orthogonality correction. It is important to remember that the wordlength errors are random variables. The computable solution of Figures 21 and 22 reflects only the expected (or mean) value of the error. It provides an estimate of the average error to be expected from a number of experiments. A simulation run is just one such experiment. The actual wordlength error determined by a simulation run will have some variation about the mean. This variation is measured by the covariance matrix P(t). For many applications, the variations are small compared to the mean value.

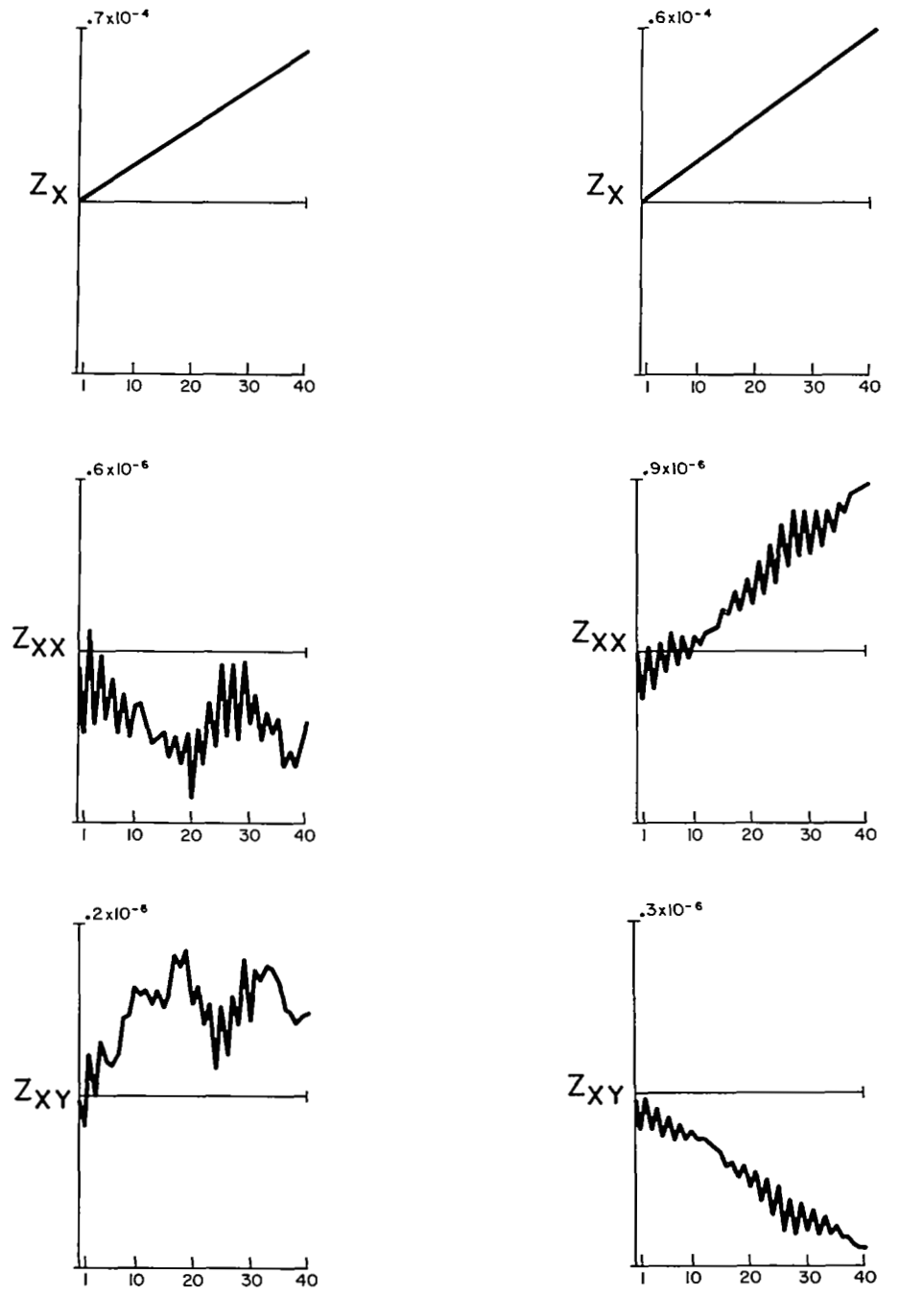


Figure 21.- One's complement, SPSA, 1108

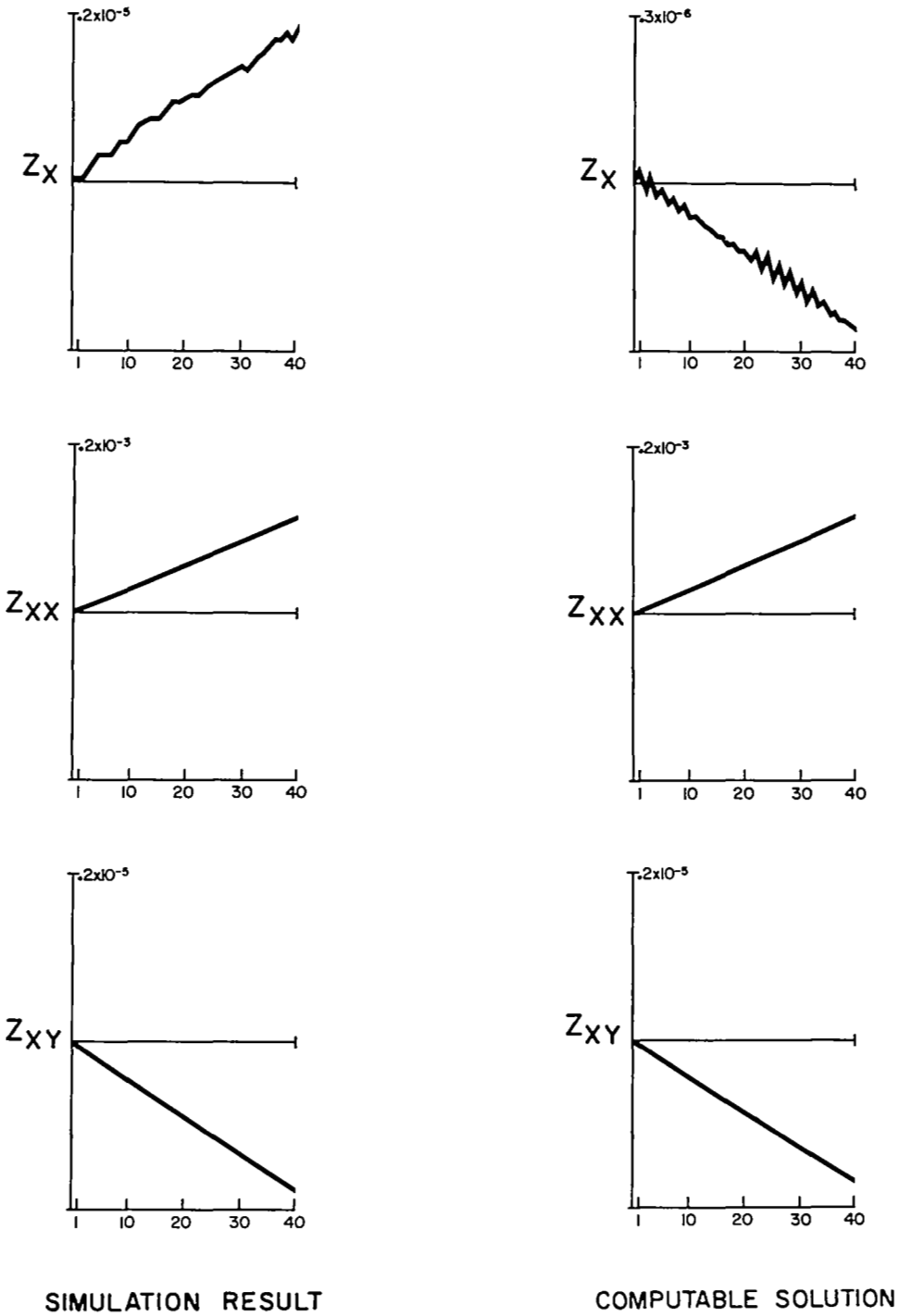


Figure 22.- Sign and magnitude, SPLA-7094

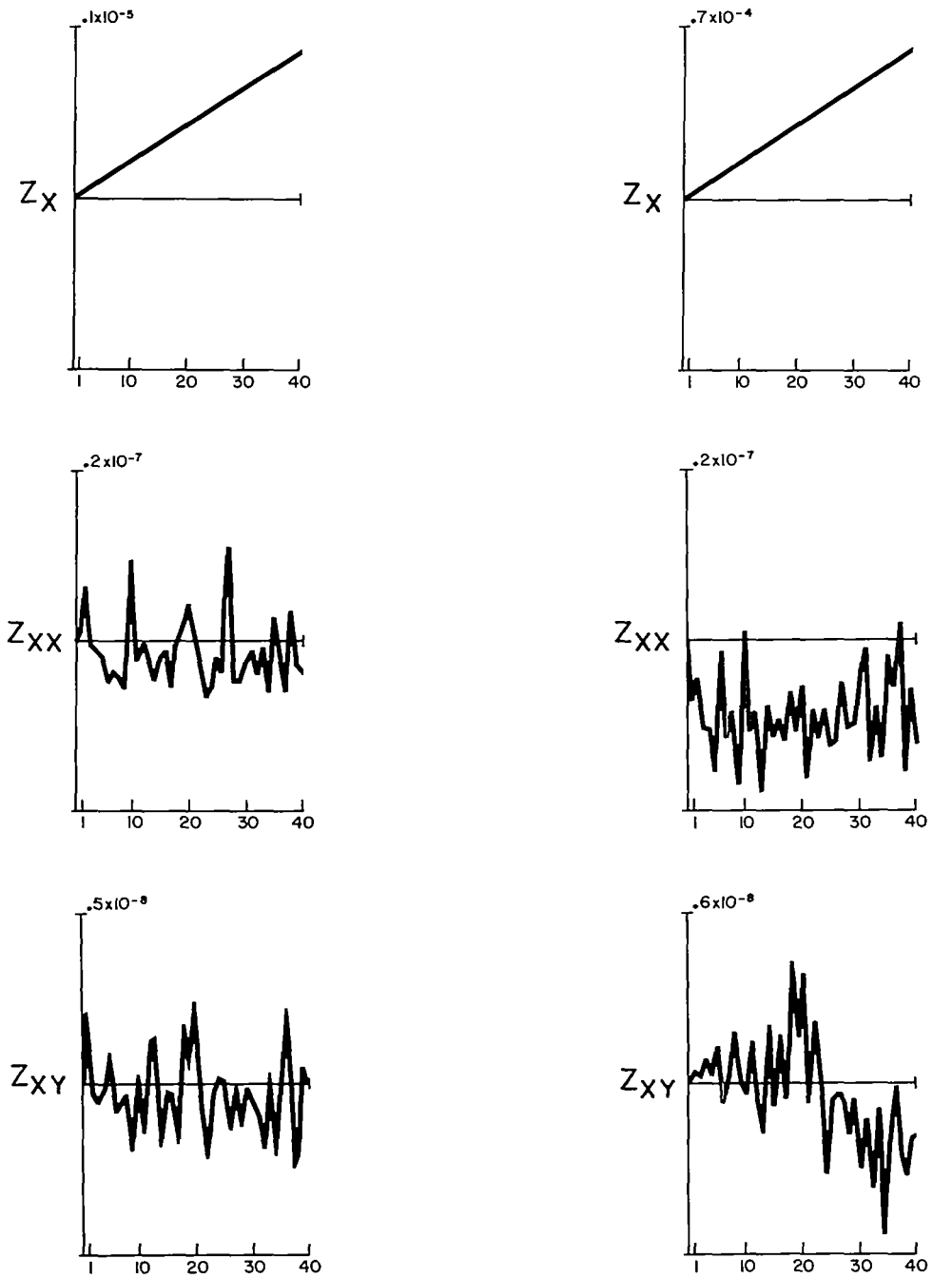


Figure 23.- Continuous orthogonality correction

TABLE XVI

TERMINAL VALUES OF WORDLENGTH ERROR

Addition		Simulation	Computable Solution
SPSA			
Zx	*	0.605×10^{-4}	0.591×10^{-4}
Zxx		-0.250×10^{-6}	0.876×10^{-6}
Zxy		0.937×10^{-7}	-0.275×10^{-6}
SPLA			
Zx		0.181×10^{-5}	-0.264×10^{-6}
Zxx	*	0.114×10^{-3}	0.116×10^{-3}
Zxy	*	-0.177×10^{-5}	-0.169×10^{-5}

Under these conditions, each simulation run will closely approximate the mean value predicted by the computable solution. This is the case of the simulation results of Figures 21 and 22. For the secondary error and when the mean value of the symmetric error is suppressed by an orthogonality correction, the results are essentially random, indicating that the variations about the mean are no longer negligible compared to the mean.

The computable solution provides an estimate of the covariance matrix P(t) as well as an estimate of the mean value. For those applications where the standard deviation of the error is much smaller than the mean value, a reduction in the overall error can be achieved by using a symmetrical rounding procedure. The rounding procedure described previously required the addition of a small correction factor to each word before it is shortened. Additional machine time is required to round off numbers in this fashion. Reference 13 contains another rounding procedure. It consists of forcing the last bit of each word to be "1". This procedure results in a zero mean wordlength error but with a distribution range twice that shown in Figure 19. The simulations of Figures 21 and 22 did not employ a rounding operation. An example of symmetric rounding (applied to a different problem) is contained in reference 9.

IV. SYSTEM PERFORMANCE EVALUATION

Up to this point, the analysis of the direction cosine error has been concerned with the error matrices E and Z. Although the solutions provide insight into the nature of the direction cosine error and, to a limited extent, allow comparisons of the different methods of calculating the cosines, a more definite solution is desirable for evaluating system performance.

Reference Trajectory

A reference trajectory is specified as a time history of vehicle angular rates and accelerations, viz.:

a_B = vehicle acceleration in body coordinates,

a_N = vehicle acceleration in navigational coordinates,

v = vehicle velocity in navigational coordinates,

r = vehicle position in navigational coordinates,

$g(r)$ = gravitational acceleration computed from the current estimate of vehicle position,

B = true direction cosine matrix,

$\hat{B}$ = on-board estimate of the cosine matrix.

Except for the rather restricted case of a self-commutative rate matrix, a solution for the true direction cosine matrix will, in all probability, not be known. Under these circumstances, the true cosine matrix B is really a fictitious quantity. However, it is usually possible to determine a fairly accurate matrix by using a sophisticated integration technique such as the fourth-order Runge-kutta method. This matrix will be donoted by B*.

The nominal mission is "flown" by integrating the equations

$$\dot{B}^* = \Omega_B^* \tag{170}$$

$$\dot{v} = a_N + g(r)$$

$$\dot{r} = v$$

The result is a time history of nominal vehicle position and velocity. If the acceleration time history is available in body coordinates rather than navigational coordinates, the transformation

$$a_N = B^{*T} a_B \tag{171}$$

may be used. The error in the cosine matrix is of no consequence since any actual mission will not follow the nominal trajectory exactly. All that is necessary is a profile reasonably close to the nominal to use as a reference.

The effect of direction cosine errors can be evaluated by simultaneously integrating the equations

$$\delta\dot{v} = Z^T a_N + \delta g(r) \tag{172}$$

$$\delta\dot{r} = \delta v$$

where $\delta g(r)$ is determined from δr and the nominal r vector.

The Z matrix can be found (for a general angular rate input) (1) by finding an analytic W matrix. Then:

$$R = B^{*T} W B^* \tag{173}$$

where the errors in the B* matrix generate second order errors in the R matrix. The Z matrix is then found by integrating

$$\dot{Z} = RZ + R \ ,$$

or (2) by using a computable solution for the Z matrix.

For either method, it is necessary to make a number of runs for the different types of errors and algorithms. It is not necessary to make runs for different computational rates or wordlengths since these appear as parameters in the error expressions. The algorithm error and the mean value of the wordlength error may be treated as deterministic. The variation of the wordlength error about the mean must be given a one-, two-, or three-sigma value depending upon the corresponding choice for the statistical

instrument errors. The validity of the final results is highly dependent upon the choice of a realistic vehicle environment. For example, the rate matrix $\Omega(t)$ may include sinusoidal terms to represent vehicle limit cycling and angular vibration. Pseudo-random numbers can also be generated and passed through a shaping filter to model random vibrations.

The output of the error analysis program is the velocity, position, and attitude errors of the vehicle. The number of error terms can be reduced by calculating the magnitude of the velocity error from the vector components and using this as a single measure of system performance, or a figure of merit (FOM) may be based upon the midcourse correction velocity required of a spacecraft as a function of all error terms. The important point is that the same criteria be used for the software and hardware errors. The next section will outline a software design technique which will optimally allocate computing resources on the basis of the error analysis described here.

Software Design Technique

As the previous sections have indicated, it is not always a simple matter to determine the "best" direction cosine algorithm. The choice will be highly dependent upon the accuracy requirement, operating environment, and the computing resources available. These parameters influence the design of the entire software package, so the technique to be outlined may be applied to other programming areas as well as to the direction cosines.

A basic requirement of the software is that it maintain an acceptable level of computational error in a given environment. The accuracy specification is often taken as a fraction of the hardware error for the same mission. For this reason, the software error should be expressed in terms of the same figure of merit (FOM) as the hardware error.

The error for a software routine will have the form:

$$FOM = EAL + EWL$$

where

EAL = algorithm error

EWL = wordlength error

FOM = figure of merit which is to be minimized.

The individual terms have the form:

$$EWL = f_l^* f_c$$

$$EAL = k_n^*/f_c^n + k_m^*/f_c^m$$

where the k_i's are mission-dependent constants. For example, a second-order method would have one error term of the form:

$$EAL = k_A^*/f_c^4 \ .$$

For a given computational frequency, the fourth-order error will normally be much less than the second-order error. This is not a meaningful comparison, however, since a second-order algorithm is simpler and, for an equal amount of computer time, it could be executed at a higher computational frequency. This discrepancy may be resolved by comparing the error terms on the basis of an equal percentage of computer time. Let 'T_c' be the time required to make one pass through the algorithm. Then:

$$f_t = f_c T_c$$

is the fractional part of the computer time which must be devoted to the algorithm. The FOM may then be re-written as

$$FOM = k_l f_t + k_n/f_t^n + k_m/f_t^m \ .$$

Each algorithm will also occupy a fractional part of the computer memory. This may be denoted by f_m. The total cost of implementing any algorithm is then given by the complex number:

$$cost = f_t + jf_m \ .$$

The computational resources that have been allocated are

$$F_t = \sum_K f_{tK} \quad \text{and} \quad F_m = \sum_K f_{mK}$$

where the summations are over all the algorithms currently included in the software package. The resources available are

$$RES = (1 - F_t) + j(1 - F_m^k) \ .$$

The designer may now select the algorithms which minimize the error at a specified cost or he may minimize the cost required to meet an error specification. In the latter case, a possible performance index is the complex variable

$$J = \frac{F_t}{1 - F_t} + j \frac{F_m}{1 - F_m}$$

or the real variable

$$J = a_1 \frac{F_t}{1 - F_t} + a_2 \frac{F_m}{1 - F_m} \ .$$

In either case, the search to minimize the performance index will be over a discrete set of algorithms. In programming the algorithms, it will often be possible to trade-off memory requirements against computational time. Thus, one basic algorithm may have many variations. Each variation is included as a separate entity. As the software package nears completion, it may happen that almost all the available computer memory is used and F_m is almost unity. The performance index will then greatly favor routines which require little computer memory. Despite this, the total memory required may eventually exceed that which is available. In this case, another memory module must be added to the system. When this is done, the performance index may then weigh time efficiency much more heavily than memory efficiency, and a completely different choice of algorithms would be optimum. The choice of a "best" algorithm is thus complicated by the circumstances of its employment. In addition, the overall optimization will be constrained by the current extent of the software library.

Electronics Research Center
National Aeronautics and Space Administration
Cambridge, Massachusetts, September 1968
125-23-02-09

APPENDIX A

INITIAL CONDITION

The direction cosines propagate as

$$\dot{B} = \Omega B \ .$$

The initial condition may always be taken as the unit matrix without loss of generality since, for an arbitrary initial condition:

$$B(0) = B_0 \ .$$

A new matrix may be defined as

$$U = BB_0^T$$

$$\dot{U} = \dot{B}B_0^T = \Omega BB_0^T$$

$$\dot{U} = \Omega U \qquad U(0) = I$$

so that the solution for B and U differ only by a constant matrix B_0^T.

APPENDIX B

EXACT SOLUTIONS

For a self-commutative (but otherwise arbitrary) rate matrix, an approximate R matrix is easily found by using the narrow-band approximation. For the approximation to be valid, it is necessary that $w_0 h$ be a small number. Since this condition will be satisfied for a practical application, there is not much incentive for finding an exact R matrix. However, this can always be done. From Eq. (53) of the main body of text:

$$\Phi = I + \frac{s}{w_0} K + \frac{(1 - c)}{w_0^2} k^2$$

where

$$s = \sin w_0 H$$

$$c = \cos w_0 H.$$

If a first-order Taylor series algorithm is used:

$$\hat{\Phi} = I + HK$$

$$R = f_c(\Phi^T \hat{\Phi} - I)$$

or

$$R = f_c\left[H - \frac{s}{w_0} - (1 - c)H\right]K + \left(\frac{1 - c}{w_0^2} - \frac{sH}{w_0}\right) K^2.$$

This is an exact expression for the R matrix. To find a closed-form solution for the Z matrix, it is necessary to integrate the R matrix. The integration is usually straightforward for the approximate R matrix determined by the narrow-band method; however, for the exact R matrix, it may be considerably more difficult. An exception is the constant rate case when

$$H = h .$$

The resulting solution for α and β are given in the Table for both the first- and second-order Taylor series. The table also contains the narrow-band approximate solution. The exact solution reduces to the approximate solution for small values of $w_0 h$. The exact solution requires the subtraction of nearly equal numbers and for this reason, the approximate solutions are better suited for numerical computation.

For a sinusoidal input, the integration of the exact R matrix becomes more difficult since it involves integration of terms of the form

$$\sin (m \sin wt)$$

$$\cos (m \sin wt) \ .$$

The integration can be carried out by noting that

$$\cos (m \sin wt) = J_0(m) + 2J_2(m)\cos 2wt + \ldots$$

$$\sin (m \sin wt) = 2J_1(m)\sin wt + 2J_3(m)\sin 3wt + \ldots$$

This allows a series solution in terms of Bessel functions which is not restricted by the narrow-band criteria. The solution has little practical importance, however.

CONSTANT RATE CASE - TAYLOR SERIES ALGORITHMS

Order	Exact Solution		Narrow-Band Solution	
	α	β	α	β
1	$w_0\left(c - \frac{s}{w_0}\right)t$	$-w_0^2\left(\frac{1-c}{w_0^2} - \frac{s}{w_0}\right)t$	$\frac{-w_0^3}{3f_c^2}t$	$\frac{w_0^2}{2f_c}t$
2	$w_0\left(c - \frac{s}{kw_0} + w_0\frac{h_s}{2}\right)t$	$-w_0^2\left(\frac{1-c}{hw_0^2} - \frac{s}{w_0} + \frac{h_c}{2}\right)t$	$\frac{w_0^3}{6f_c^2}t$	$\frac{w_0^4}{8f_c^3}t$

APPENDIX C

EQUIVALENT INSTRUMENT ERRORS

Since the inertial system's accelerometers will have scale factor and skew (misalignment) errors and gyroscope errors may be expected to cause a drift of the navigational reference frame, it would be helpful if the computational (direction cosine) errors could be related to equivalent instrument errors. One reason for this approach is that the effects of the instrument errors on the position and velocity must be determined anyway, and so they may be presumed to be already known. A second reason is that, from a system standpoint, the computational error must be balanced against system hardware performance. Therefore, it might be convenient to describe the performance of the computer with the same indexes that are used for the inertial instruments without reference to a specific mission. Unfortunately, the analogy between cosine errors and instrument errors is somewhat tenuous. Let

SF = the diagonal matrix of accelerometer scale factor errors,

SK = the symmetric matrix of accelerometer misalignment angles, and

D = the matrix representing the integral of the system gyroscope drift rates

Then:

1. Since the accelerometer misalignments result from construction inaccuracies, the SK matrix would be expected to be neither symmetrix nor skew-symmetric and to contain six independent error angles. However, any matrix may be resolved into symmetric and skew-symmetric components. The skew-symmetric part of the SK matrix may be compensated by the initial alignment of the inertial system leaving SK symmetric with only three independent angles.

2. The integral of the gyro drift rate is used for D since the analysis of the direction cosine matrix led to the definition of Z and E matrices which are angular error matrices.

The acceleration error due to the instrument errors defined above will be

$$\tilde{a}_\beta = (I + D + SI)a_\beta$$

where

$$SI = SF + SK$$

$$\delta a'_N = B^T(D + S)a_\beta$$

$$\delta' a_N = B^T(D + S)Ba_N \ .$$

The computation error is given by

$$\delta a_N = Z^T a_N \ ,$$

so that an equivalence is given by

$$D + S = BZ^T B^T \ .$$

The above equation represents a congruence transformation of the Z matrix which results from the fact that the instrument error is defined in the body coordinate frame while the Z matrix (representing the computational error) is defined in the navigational coordinate frame. In general, the computational and instrument errors are related by the time varying B matrix. An equivalence can now be defined by

$$D = B(-Z_{ss} - Z^T Z)B^T$$

$$S = B(Z_s + Z^T Z)B^T \ .$$

When an orthogonality correction is used, the skew error will be

zero. When the angular rate matrix is self-commutative:

$$D = -Z_{ss} - Z^T Z$$

$$S = Z_s + Z^T Z \ .$$

The D matrix can be approximated by

$$D = -Z_{ss}$$

which is skew symmetric.

It is common practice to assume a constant gyro drift term (D will then be a linear function of time) and a constant S matrix. However, Z may be a more complex function of time. Therefore, any equivalence between instrument and computational errors can only be meaningful in the context of equivalent instrument coefficients which are allowed to be functions of time.

APPENDIX D

EFFICIENT COMPUTATION

The algorithms used in this report were written in matrix form which is convenient for analysis purposes. The programming of the equations, on the other hand, is often considerably simplified if it is not done in matrix form.

With the skew-symmetric matrix of gyroscope inputs θ [see Eq. (62)] a three-dimensional vector ψ may be associated where ψ contains the three independent elements of the matrix θ. From Eq. (95) the exact transition matrix is seen to have the form

$$\Phi = I + a\theta + b\theta^2 \quad . \tag{D-1}$$

This can be written as

$$\Phi = I + \theta_A + \theta_B\theta \tag{D-2}$$

where

$$\psi_A = a\psi \tag{D-3}$$

$$\psi_B = b\psi$$

with the modification of ψ applied to a vector rather than to a matrix.

Approximate transition matrices have the same general form. From Eqs. (66) and (95), it is evident that a fourth-order approximation has the form of Eq. (D-2) provided that

$$a = 1 - \phi^2/6 \tag{D-4}$$

$$b = 1/2 - \phi^2/24$$

where ϕ^2 is defined by Eq. (95).

The same approach can be applied to the compensated solution of Eq. (120). Two gyroscope samples are obtained (ψ_1, ψ_2) for each calculation of a compensated cosine matrix. Eq. (120) can then be approximated by

$$\psi = \psi_1 + \psi_2 + \frac{2}{3}(\psi_1 \times \psi_2). \qquad \text{(D-5)}$$

The sequence of calculations is then Eqs. (D-5), (D-4), (D-3), and (D-2). In addition Eq. (D-2) need not be programmed in matrix form. The result is a fourth-order algorithm with a commutativity correction incorporated in Eq. (D-5).

REFERENCES

1. Martin, J.F.P.: Some Results on Matrices Which Commute with Their Derivatives. SIAM Journal of Applied Math., Vol. 15, No. 5, September 1967.

2. TRW Systems: Body-Fixed, Three-Axis Reference System Study, Phase II, Final Report. Prepared for NASA George C. Marshall Space Flight Center, Huntsville, Alabama, under Contract No. NAS8-20209, 15 December 1966.

3. United Aircraft Corporate Systems Center: A Study of the Critical Computational Problems Associated with Strapdown Inertial Navigational Systems. NASA CR-968, April 1968.

4. Sullivan, J.J.: Evaluation of the Computational Errors of Strapdown Navigational Systems. AIAA Journal, Vol. 6, No. 2, February 1968.

5. Wilkinson, J.H.: Rounding Errors in Algebraic Processes. Prentice-Hall, New Jersey, 1963.

6. Henrici, P.: Error Propagation for Difference Methods. John Wiley and Sons, New York, 1963.

7. Hull, T.E., and Swenson, J.R.: Tests of Probabilistic Models for Propagation of Roundoff Errors. Communications of the ACM, Vol. 9, No. 2, February 1966.

8. Errors in Digital Computation. Vol. 1, edited by L.B. Rall, John Wiley and Sons, New York, 1964.

9. IBM: Computer Precision Study II, Report No. SSD-TDR-65-134. Prepared for U.S. Air Force Space Systems Division, Air Force Systems Command, Los Angeles, Calif., under Contract AF04(695)-725, October 1965.

10. Lee, J.S., and Jordan, J.W.: The Effect of Word Length in the Implementation of an On-Board Computer. Technical paper presented at the Institute of Navigation, National Space Navigation Meeting, Los Angeles, Calif., March 1967.

11. Schwarz, R.J., and Friedland, B.: Linear Systems. McGraw-Hill Book Company, New York, 1965.

12. Elgerd, O.I.: Control Systems Theory. McGraw-Hill Book Company, New York, 1967.

13. Chu, Y.: Digital Computer Design Fundamentals. McGraw-Hill Book Company, New York, 1962.

DIRECTION COSINE COMPUTATIONAL ERROR

By John W. Jordan
Electronics Research Center
Cambridge, Massachusetts

ABSTRACT

Strapdown inertial systems replace the gimbal structure of more conventional inertial systems by a direction cosine matrix which is contained in the system computer. Since the cosine matrix must be updated many times a second, a detailed study of the computational error introduced by the on-board computer is necessary to determine system performance. This report is concerned with the techniques which may be employed to estimate the computational error and its effects on system performance. Although focused upon a particular problem of practical importance, many of the techniques have a wider applicability to aerospace software in general. The computer-generated error is divided into two categories: (1) Algorithm Error which is caused by the approximate nature of the numerical formulas used, and (2) Wordlength Error which is caused by a finite computer wordlength.

Wordlength Error is treated from the statistical viewpoint of Henrici. The theory is developed for linear matrix differential equations. The direction cosine problem is then used as a specific example. The results of Henrici are extended to include different computer number systems (i.e., sign-magnitude or complement representations) and different organizations of the computer's arithmetic unit. The emphasis is upon a computable solution which may be applied to problems for which analytical solutions are either not possible or are not practical. The solution includes both fixed- and floating-point machines. The computable solution is verified by simulation.

A differential equation is derived for the propagation of the Algorithm Error. Since the direction cosine matrix should be orthogonal, it is possible to include a correction routine in the airborne computer so that the cosine matrix will remain orthogonal despite computational error. A second differential equation is derived for the propagation of the Algorithm Error when an orthogonality correction is used. Closed-form analytical solutions are obtained for both error differential equations provided that the vehicle angular rate matrix is self-commutative.

In the general case, the error is determined by a computable solution or by simulation. The report also considers compensation

techniques, the evaluation of system performance, and the "optimum" design of the aerospace software. All analytical solutions are verified by simulation.

STAR Categories 08, 19, and 21