TECHNICAL NOTE NO. 17

Research Project A-588

SAMPLING ERRORS IN CLOSED LOOP HYBRID COMPUTER PROGRAMS

By Joseph L. Hammond, Jr.

Prepared for
George C. Marshall Space Flight Center
Huntsville, Alabama

Contract No. NAS8-2473

(Development of New Methods and Application of Analog Computation)

15 May 1968

School of Electrical Engineering
GEORGIA INSTITUTE OF TECHNOLOGY
Atlanta, Georgia

GEORGIA INSTITUTE OF TECHNOLOGY
School of Electrical Engineering
Atlanta, Georgia 30332

(GIT/EES Report A588/T17)

15 May 1968

SAMPLING ERRORS IN CLOSED LOOP HYBRID COMPUTER PROGRAMS

By

Joseph L. Hammond, Jr.

TECHNICAL NOTE NO. 17

on

Contract No. NAS8-2473

(Development of New Methods and Applications of Analog Computation)

For

GEORGE C. MARSHALL SPACE FLIGHT CENTER

Huntsville, Alabama

ABSTRACT

A vector differential equation for the error due to sampling in closed loop hybrid computer programs is developed using several approximations derived with Taylor series expansions. The equation is applicable to general hybrid programs with no restriction on the manner in which the computing operations are allocated between the analog and digital computers. The major restrictions necessary for the error equation to apply are:

(1) the problem equation must be expressible as a set of first order (linear, nonlinear or time varying) equations,

(2) the digital-to-analog converters must be zero-order hold,

(3) all converters and numerical methods of the digital computer must have the same sampling period,

(4) all digitally generated functions must be computed during the same time period and converted D-to-A at the same time and finally,

(5) the sampling period must be small.

The error equation is linear. Its homogeneous part is independent of the allocation of operations between the analog and digital computers, but its forcing function depends on the details of such allocation and certain constants, namely: the sampling period, the digital execution time and the order of numerical methods used in the digital computer. Both parts of the error equation depend on the problem solution variables, but these can be either the true solutions or the actual hybrid computer solutions.

Solution of the error equation typically requires machine computation, but several properties of sampling error are apparent from the form of the equation. For example, since the forcing function on the error equation is proportional to the first power of the sampling period, it follows that the hybrid computer is

a first order computational method. It is also apparent from the error equation that the execution time of digitally generated functions has the same general effect as a non-zero sampling period but weighted twice as heavily.

The error equation is expected to be useful in studying existing hybrid computer programs, in allocating computing operations between the analog and digital computers and in compensating against sampling error.

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY OF PRINCIPAL SYMBOLS

$u(t)$ - ideal vector solution of problem equation

$x(t)$ - ideal vector solution of equation programmed on the analog computer

$z(t)$ - ideal vector solution of equation solved digitally

h - vector of functions equated to $\dot{u}(t)$

f - vector of functions equated to $\dot{x}(t)$

f_a - vector of functions generated and used in the analog computer

f_d - vector of functions generated digitally for use in the analog computer

g - vector of functions equated to $\dot{z}(t)$

t - time

n - order of vectors

p - order of numerical method

φ - principal error function of numerical method

δ - sampling period

e - execution time of digital computer

k - index of discrete time

ϵ - discretization error in numerical method

$\Gamma(t)$ - total hybrid computer sampling error vector

$\gamma(t)$ - sampling error vector for analog computer

$\alpha(t)$ - sampling error vector for digital computer

F_d - output of D/A converter with f_d as input

$Z(t)$ - output of D/A converter with $z(t)$ as input

$u^*(t)$ - vector solution of problem equations which accounts for hybrid computer sampling error (similar notation for other variables)

GLOSSARY OF PRINCIPAL SYMBOLS (CONT'D.)

$\partial h/\partial u$ - a matrix defined in the Appendix (a similar notation for other variables)

$O[g(\delta)]$ - a symbol denoting that some variable (which is $O[g(\delta)]$) goes to zero with δ at least as rapidly as $g(\delta)$

$$u(t) = \begin{bmatrix} x(t) \\ z(t) \end{bmatrix}$$

$$h = \begin{bmatrix} f_a + f_d \\ g \end{bmatrix}$$

$$f = f_a + f_d$$

$$\Gamma(t) = \begin{bmatrix} \gamma(t) \\ \alpha(t) \end{bmatrix}$$

INTRODUCTION

The work reported herein is a part of a general study of sampling errors in hybrid computation being carried out as one of several tasks under the current research program. Earlier work on the same problem has been reported in Technical Notes Nos. 8, 12, and 13.[1],[2],[3]† In these reports a differential equation for sampling error in certain hybrid computer applications is derived and evaluated in test examples.

The objective of the present technical note is to document further study which has unified and generalized earlier work. Specifically the present document derives a differential equation for the sampling error in a general class of hybrid computer applications. The results apply to any allocation of computing tasks between the analog and digital equipment and take into account errors in the numerical techniques used in the digital computer. The restrictions on the present analysis are listed on pages 5 and 6 in the body of the report. The most important restrictions are:

(a) The implicit requirement that the problem equation can be expressed in "state variable" form (given as (1) on page 3).

(b) The assumption that the D/A converters are zero-order hold.

(c) The assumption that all converters and the numerical method have the same sampling period.

(d) The assumption that all digitally generated functions are converted D-to-A at the same time after a common execution time.

---

†Numbers refer to references listed in the Bibliography.

These assumptions are much less restrictive than those used in Technical Notes Nos. 8, 12, and 13 since the earlier work applies only in applications for which the digital computer serves as a digital function generator. It can be noted, however, that the general sampling error equations derived in this technical note reduce to those of the earlier notes in the special cases for which the earlier work applies.

In order to place clearly in evidence the underlying assumptions leading to the sampling error equation, the approach in this report is to develope equations for the hybrid computer system including the effect of sampling error. Using this result an equation for the sampling error, defined as the difference between an ideal solution of the problem equation and a hybrid solution accounting for sampling error, is derived. The technical note then examines the form assumed by the error equation in several special cases and gives a numerical example illustrating a general hybrid computer implementation.

DEVELOPMENT OF HYBRID COMPUTER EQUATIONS

In order to obtain tractable results it will be assumed that the hybrid computer is programmed to solve the ideal equation

$$\dot{u}(t) = h\{u(t);t\} \ . \tag{1}$$

The u(t) is an n-vector† of component outputs.

Equation (1) is implemented on the hybrid computer as shown in Figure 1, where the double lines represent the flow of vector quantities. The ideal equation (1) can be partitioned to place in evidence the ideal equation solved by the analog computer and the ideal equation solved by the digital computer. The result is

$$\dot{x}(t) = f_a\{x(t),z(t);t\} + f_d\{x(t),z(t);t\} \tag{2}$$

and

$$\dot{z}(t) = g\{x(t),z(t);t\} \ , \tag{3}$$

where (2) is integrated by the analog computer to yield x(t), (3) is integrated by the digital computer to yield z(t), f_a is a vector function generated in the analog computer, f_d is a vector function generated in the digital computer,

$$u = \begin{bmatrix} x \\ z \end{bmatrix} \text{ and } h = \begin{bmatrix} f_a + f_d \\ g \end{bmatrix}$$

†Throughout the report variables without numerical subscripts are vector quantities, whereas variables with numerical subscripts are scalar quantities.

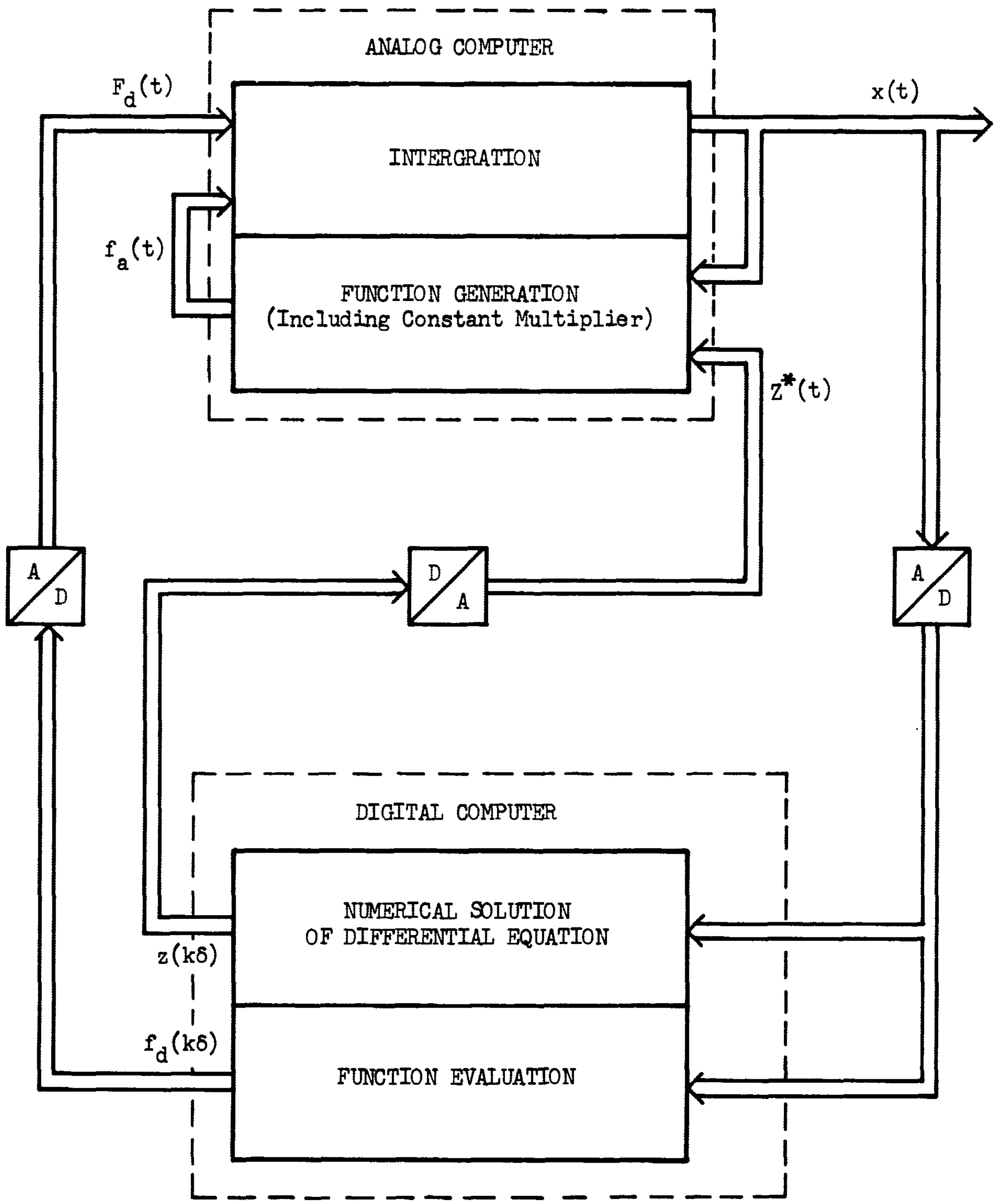

Figure 1. General Hybrid Computer Implementation of Equation (1).

Note that the hybrid computer is used in a general way with the exception that no functions are generated with the analog equipment for use in the digital computer. Such a possibility is not included for two reasons, namely, it seldom affords any practical advantage and the sampling errors occuring when the analog equipment generates functions for use in the digital computer are no different than when such functions are produced in the digital computer.

Since (2) and (3) are ideal equations, the next step is to formulate equations which account for the sampling errors present in actual hybrid equipment. The following assumptions concerning the operation of the equipment are made:

(a) The only sources of error are the non-zero sampling period of the A/D converters, and the discretization error of the numerical method used by the digital computer.

(b) The numerical method used by the digital computer to solve (3) is a method of order p with principal error function φ.

(c) The D/A converters are zero-order hold so that their outputs, F_d or Z, are stair-step functions of time.

(d) The vector of variables $z(k\delta)$ is computed digitally prior to the time $t = k\delta$ when its components are sampled by the D/A converters. This assumption requires that the execution time, necessary in computing $z(k\delta)$ from values of $z[(k-1)\delta]$ and $x[(k-1)\delta]$, be less than δ.

(e) The sampling periods of all of the converters and the step size of the numerical methods are all equal and denoted δ.

(f) Finally all components of the function $f_d[x,z;k\delta]$ are computed in a time interval e from values of x and z at $k\delta$ and all are converted D-to-A at $t = k\delta + e$.

While the assumptions are made to make the problem tractable, they are all reasonable. Assumption (a) is made to isolate one type of error for consideration. Assumption (b) limits the numerical method of the digital computer to one for which "order" and "principal error function" are useful properties. Assumption (c) and (d) are satisfied in almost all hybrid computer applications. Assumption (e), that all converters and the numerical method have a period δ, is restrictive but is a condition frequently used in practice. Finally, assumption (f), while restrictive, corresponds to one fairly common mode of operation.

Non-ideal equations corresponding to (2) and (3) and to the interconnection diagram of Figure 1 can now be obtained by expressing the assumptions listed above mathematically. Since any error will cause the solution of the non-ideal equations to differ from that of the ideal equations, the non-ideal hybrid variables will be denoted x*t(t) and z*(t).

Assumption (c) results in the expressions

$$Z^*(t) = z^*(i\delta),\ i\delta \le t < (i+1)\ \delta \qquad (4)$$

and

$$F_d(t) = f_d\ \{x^*(i\delta), z^*(i\delta); i\delta\},\ i\delta + \le t < (i+1)\delta + e \qquad (5)$$

for $Z^*(t)$ and $F_d(t)$, the outputs of the D/A converters. Note that execution time e appears in (5) in accordance with assumption (f). Equations (4) and (5) along with Figure 1 can be used to express the equation solved by the analog computer as

$$\dot{x}^*(t) = f_a\{x^*(t), Z^*(t); t\} + F_d(t) \ . \tag{6}$$

The ideal equation to be solved by the numerical method in the digital computer is (2) with $x(t)$ replaced by $x^*(t)$. Because of this change the solution of the ideal equation is no longer $z(t)$, and (2) is written as

$$\dot{z}'(t) = g\{x^*(t), z'(t); t\} \tag{7}$$

A relation between $z'(t)$, the true solution of (7), and $z^*(i\delta)$, the numerical solution of (7), must now be obtained. In numerical analysis, sampling error is typically referred to as "discretization" error, and in keeping with the general assumptions other errors introduced in the numerical solution of (7) will be neglected. Many different algorithms for the numerical solution of (7) exist. For most of these, the so-called accumulated discretization error given by

$$\epsilon(i\delta) = z^*(i\delta) - z'(i\delta), \ i = 0,1,2 \ . \ . \ . \tag{8}$$

can be approximated asymptotically as the solution to an equation of the form†

$$\dot{\epsilon}(t) = \frac{\partial g}{\partial z'} \epsilon(t) + \delta^P \varphi(z'; t); \ \epsilon(0) = 0 \tag{9}$$

---

†Such equations are discussed for example by Henrici [4]. For (9) to apply for Runge-Kutta type algorithms the function g must be evaluated at several points internal to the basic $[i\delta, (i+1)\delta]$ interval. This requires that a forcing function implicit in g must be sampled at a rate higher than $1/\delta$. This does not pose a problem if digital execution time rather than characteristics of the D/A and A/D converters are assumed to determine the minimum sampling period.

For predictor-corrector type algorithms (9) is more approximate than for one-step or Runge-Kutta type algorithms since it neglects error in starting predictor-corrector procedures.

where g is the function on the right-hand side of (7), p is the order of the numerical method and the accumulated discretization error at $t = i\delta$ is $\epsilon(t{=}i\delta)$. The function φ is termed the principal error function, and in all cases for which (9) applies expressions for it are available. The symbol $\partial g/\partial z'$ with g and z' both vectors is defined in the Appendix. Note that (9) is a linear equation whose solution can be expressed as

$$\epsilon(t) = \delta^{p}\ \epsilon_{o}(t)$$

where $\epsilon_o(t)$ is the solution of (9) with $p = 0$. Hence if $\epsilon_o(t)$ is bounded as will be assumed, for small δ $\epsilon(t)$ is $O(\delta^p)$.†

It is useful to introduce a continuous variable $z^*(t)$ by the definition

$$z^*(t) = z'(t) + \epsilon(t) \ . \tag{10}$$

Here $z'(t)$ is the true solution of (7) and $\epsilon(t)$ is obtained as the solution of (9). An inspection of (8) and (10) shows that $z^*(t)$ is the digital computer output at the discrete time instants $t = i\delta$, $i = 0, 1, 2, \ldots$ A differential equation for $z^*(t)$, namely

$$\dot{z}^*(t) = g\{x^*(t), z'(t); t\} + \frac{\partial g}{\partial z'}\ \epsilon(t) + \delta^{p}\varphi(z';t), \tag{11}$$

results from differentiating (10) and using (7) and (9).

Equation (11), which involves the variables $z^*(t)$, $z'(t)$, and $\epsilon(t)$ can be expressed in terms of only the first of these variables through use of the general approximations developed in the Appendix. Consider the first

---

†The symbol $O(\delta^p)$ has the following significance. If $f(\delta)$ is $O[g(\delta)]$ as $\delta \to 0$, then there exists a positive constant c such that $|f(\delta)| \le c|g(\delta)|$ for δ sufficiently close to zero.

two terms on the right-hand side of (11), namely $g\{x^*(t), z'(t); t\} + \frac{\partial g}{\partial z'} \epsilon(t)$. Use of (A-2)[†] shows that for small $\epsilon(t)$ this sum is approximately equal to $g\{x^*(t), z'(t) + \epsilon(t); t\}$, or using (10), to $g\{x^*(t), z^*(t); t\}$. Similar reasoning results in the expression

$$\varphi(z'; t) = \varphi\{z^*(t) - \epsilon(t); t\} = \varphi\{z^*(t); t\} - \frac{\partial \varphi}{\partial z^*} \epsilon(t). \quad (12)$$

Using these expressions for g and φ in (11) then yields

$$\dot{z}^*(t) = g\{x^*(t), z^*(t); t\} + \delta^P \varphi\{z^*(t); t\} \quad (13)$$

after neglecting the term $\delta^P \frac{\partial \varphi}{\partial z^*} \epsilon(t)$ which is $O(\delta^{2P})$.

Equations (13) and (6) are the equations effectively solved by the hybrid computer system accounting for sampling error but assuming that δ is small.

[†]Equations numbered (A-) appear in the Appendix.

DEVELOPMENT OF SAMPLING ERROR EQUATIONS

A sampling error vector $\Gamma(t)$ can be defined as the difference between the solution of the ideal equations and the hybrid computer equations which account for sampling error. Thus

$$\Gamma(t) = \begin{bmatrix} \gamma(t) \\ \alpha(t) \end{bmatrix} = \begin{bmatrix} x(t) - x^*(t) \\ z(t) - z^*(t) \end{bmatrix} \tag{14}$$

where $\Gamma(t)$ is the total hybrid computer error vector and $\gamma(t)$ and $\alpha(t)$ are, respectively, the analog and digital computer error vectors. Differentiation of (14) yields

$$\dot{\Gamma}(t) = \begin{bmatrix} \dot{\gamma}(t) \\ \dot{\alpha}(t) \end{bmatrix} = \begin{bmatrix} \dot{x}(t) - \dot{x}^*(t) \\ \dot{z}(t) - \dot{z}^*(t) \end{bmatrix} . \tag{15}$$

A differential equation for error is obtained by subtracting (6) from (2) and (13) from (3). The result is

$$\dot{\gamma}(t) = f_a\{x(t),z(t);t\} - f_a\{x^*(t),Z^*(t);t\} + f_d\{x(t),z(t);t\} - F_d(t) \tag{16}$$

$$\dot{\alpha}(t) = g\{x(t),z(t);t\} - g\{x^*(t),z^*(t);t\} - \delta^P\varphi\{z^*(t);t\} . \tag{17}$$

Use is now made of the general approximations developed in the Appendix to reduce (16) and (17) to a more tractable form by neglecting higher order terms in the variables δ, α and γ which are assumed to be "small." Specifically, terms $O(\delta^2)$, $O(\gamma^2)$, $O(\alpha^2)$ and $O(\alpha\delta)$ are neglected.

First consider the staircase variables $F_d(t)$ and $Z^*(t)$. Using (A-6) and (A-8) these can be approximated as

$$Z^*(t) \simeq z^*(t) - [\delta/2 + S(t)]\dot{z}^*(t) \tag{18}$$

$$F_d(t) \simeq f_d\{x^*(t),z^*(t);t\} - \left[\frac{\delta+2e}{2} + S(t-e)\right]\dot{f}_d\{x^*(t),z^*(t);t\} . \tag{19}$$

Using (18) and (19) in (16) then yields

$$\dot{\gamma}(t) = f_a\{x(t),z(t);t\} - f_a\{x^*(t),z^*(t) - [\delta/2 + S(t)]\dot{z}^*(t);t\} \tag{20}$$

$$+ f_d\{x(t),z(t);t\} - f_d\{x^*(t),z^*(t);t\} + \left[\frac{\delta+2e}{2} + S(t-e)\right] f_d\{x^*(t),z^*(t);t\} .$$

Equation (A-2) is then used to modify (20) and (17) to the extent that all of the functions on the right-hand side of these equations have as argument the non-ideal hybrid variables $x^*(t)$ and $z^*(t)$. For example,

$$f_a\{x(t),z(t);t\} = f_a\{x^*(t),z(t);t\} + \frac{\partial f_a}{\partial x^*}\gamma(t) + \frac{\partial f_a}{\partial z^*}\alpha(t) .$$

The result of this step after canceling certain terms is

$$\dot{\gamma}(t) \simeq \frac{\partial f_a}{\partial x^*}\gamma(t) + \frac{\partial f_a}{\partial z^*}\alpha(t) + \frac{\partial f_d}{\partial x^*}\gamma(t) + \frac{\partial f_d}{\partial z^*}\alpha(t) + S(t)\frac{\partial f_a}{\partial z^*}\dot{z}^*(t) + \frac{\delta}{2}\frac{\partial f_a}{\partial z^*}\dot{z}^*(t)$$

$$+ \left(\frac{\delta+2e}{2}\right)\dot{f}_d\{x^*(t),z^*(t);t\} + S(t-e)\,\dot{f}_d\{x^*(t),z^*(t);t\} \tag{21}$$

$$\dot{\alpha}(t) \simeq \frac{\partial g}{\partial x^*}\gamma(t) + \frac{\partial g}{\partial z^*}\alpha(t) - \delta^P\varphi\{z^*(t);t\} . \tag{22}$$

In these equations and the error equations below $\alpha(t=0) = \gamma(t=0) = 0$.

In (21) note that the two terms involving S(t) are forcing functions for the linear differential equation. Thus, assuming the conditions of approximation 3 in the Appendix are met, the contributions of these terms to $\gamma(t)$ will be $O(\delta^2)$ and hence they can be neglected.

Each term on the right-hand side of (21) and (22), after neglecting the terms involving S, is multiplied by one of the small quantities γ, α, or δ.† Thus an application of (A-2) to change the arguments of the functions on the right-hand side of (21) and (22) to the ideal variables x(t) and z(t) will result in added terms which are all of higher order in the small quantities. For example, the term $\frac{\partial f_d}{\partial x^*}\gamma(t)$ can be expressed as

$$\gamma(t)\,\frac{\partial f_d}{\partial x^*}\{x^*(t),z^*(t);t\} = \gamma(t)\,\frac{\partial f_d}{\partial x}\{x(t),z(t);t\} - \frac{\partial^2 f_d}{\partial x^2}\gamma^2(t) - \frac{\partial^2 f_d}{\partial z\partial x}\alpha\gamma$$

and the last two terms are $O(\gamma^2)$ and $O(\alpha\gamma)$, respectively. Thus (21) and (22) can be written as

$$\dot{\gamma}(t) \simeq \frac{\partial f_a}{\partial x}\gamma(t) + \frac{\partial f_a}{\partial z}\alpha(t) + \frac{\partial f_d}{\partial x}\gamma(t) + \frac{\partial f_d}{\partial z}\alpha(t) + \frac{\delta}{2}\frac{\partial f_a}{\partial z}g + \left(\frac{\delta+2e}{2}\right)\dot{f}_d \quad (23)$$

and

$$\dot{\alpha}(t) \simeq \frac{\partial g}{\partial x}\gamma(t) + \frac{\partial g}{\partial z}\alpha(t) - \delta^P\varphi \quad (24)$$

† All practical numerical methods have $p \geq 1$ since this is necessary for convergence of the numerical solution to the ideal solution as δ approaches zero.

where to the accuracy being considered the arguments of the functions can be either the ideal or the non-ideal hybrid solution variables and time.† In obtaining (23), (3) is used to replace $\dot{z}(t)$ with $g\{x(t),z(t);t\}$.

Equations (23) and (24) represent the most general form of the equations for sampling error to be considered in this report. Note that the equations are linear but can be time varying if the partial derivatives of f_a, f_d or g depend on time.

Using the previously defined quantities h and Γ, (23) and (24) can be written more compactly as

$$\dot{\Gamma}(t) = \frac{\partial h}{\partial u}\Gamma(t) + \begin{bmatrix} \delta/2 \dfrac{\partial f_a}{\partial z} g + \dfrac{\delta+2e}{2} \dot{f}_d \\ - \delta^p \varphi \end{bmatrix}. \qquad (25)$$

Note that the homogeneous part of the sampling error equation depends only on the total function h, which is independent of how the equations are divided between the analog and digital computers in the hybrid implementation. The function h in turn depends on either the ideal solutions or the hybrid solutions and on time. The forcing function for (25), on the other hand, depends on how the equations are divided between the analog and the digital computers (i.e., on f_d and g). It also depends on the principal error function φ of the numerical method, either the ideal or hybrid solution variables and on the constants δ, e and p.

---

†Such arguments will be understood below unless otherwise stated. As a further change in notation the approximate equality sign $\simeq$ will be replaced by the standard equality sign.

For purposes of solving the error equation numerically, it may be expedient to use the chain rule to express $\dot{f}_d$ in (25) as

$$\frac{df_d}{dt} = \frac{\partial f_d}{\partial z} g + \frac{\partial f_d}{\partial x} f + \frac{\partial f_d}{\partial t} \qquad (26)$$

where $\frac{\partial f_d}{\partial t}$ accounts for the explicit dependence of f_d on t. With the possible use of the result of (26), (25) can be programmed for a general purpose computer. The required vector operations can be performed by the computer whose only inputs can be a specification of the functions f_a, f_d, g and φ; either the ideal or hybrid solutions of the given equation (i.e x(t),z(t) or x*(t) and z*(t)) and the constants δ, e and p.† It is, of course, possible to closely approximate x(t) and z(t) by also solving (2) and (3) when using a digital computer and thus remove the necessity for these functions as inputs. Another possibility would be to solve (25) on the hybrid computer along with (2) and (3). In this case x*(t) and z*(t) would be used in (25).

†The work of Moore [5] and Reiter [6] may provide an efficient and accurate method for machine generation of the required partial derivatives.

THE ERROR EQUATION FOR CERTAIN SPECIAL CASES

Certain more specific assumptions concerning the given equation or the method of hybrid implementation result in simplifications of the general error equation.

Hybrid Implementation with Zero Sampling Error

Examination of (25) shows that in certain cases (which are trivial from a practical point of view) the general error equation reduces to a homogeneous equation with zero initial conditions so that the approximate error vectors $\alpha(t)$ and $\gamma(t)$ are zero for all time. This occurs for two cases, namely: (1) the hybrid computer has δ identically zero or (2) the following conditions apply $-f_a$ is independent of z (hence $\partial f_a/\partial_z = 0$), f_d is zero (hence $\dot{f}_d \equiv 0$), and p is large (hence δ^P is negligibly small).

That error is zero in both of these cases is intuitively obvious. Case 1 requires no further comment. Case 2 is just the condition for the equations for $\gamma(t)$ and $\alpha(t)$ to be independent and the error in the numerical method negligible.

Hybrid Implementations with Negligible Error in the Numerical Method

Consider the term $\delta^P\varphi$ in (25). This term which accounts for error in the numerical method is $0(\delta^P)$ and hence will be negligible for $p \geq 2$ if φ is bounded. For such cases the error equation becomes

$$\dot{\Gamma}(t) = \frac{\partial h}{\partial u}\Gamma(t) + \begin{bmatrix} \delta/2\,\dfrac{\partial f_a}{\partial z}\,g + \dfrac{\delta+2e}{2}\,\dot{f}_d \\ 0 \end{bmatrix}.$$

This equation applies in almost any case for which sophisticated numerical methods are used in the digital computer.

Hybrid Implementations Which are Predominately Analog or Digital

There are hybrid implementations for which either the analog or the digital computations are trivial in the sense that only algebraic rather than differential equations are solved. In such cases, the given equations are integrated exclusively on either the analog or the digital computer. Equation (25) then reduces to either

$$\dot{\gamma}(t) = \frac{\partial f}{\partial x} \gamma(t) + \frac{\delta+2e}{2} \dot{f}_d \tag{27}$$

or

$$\dot{\alpha}(t) = \frac{\partial g}{\partial z} \alpha(t) - \delta^{p}\varphi \ , \tag{28}$$

where $f = f_a + f_d$.

Equation (27) applies in the relatively important case of a hybrid implementation which uses the digital subsystem entirely as a function generator.†

Equation (28) applies, when the analog computer is used exclusively as an "algebraic function generator" for the digital computer. Such implementations are not in common use.

Hybrid Implementations with Uncoupled Analog and Digital Computations

Examination of (23) and (24) indicates that the equation for $\gamma(t)$ is coupled to the equation for $\alpha(t)$ by terms multiplied by $\frac{\partial f}{\partial z}$. Similarly, the equation for $\alpha(t)$ is coupled to the equation for $\gamma(t)$ by terms multiplied by $\frac{\partial g}{\partial x}$. Thus for systems with $\frac{\partial f}{\partial z} \equiv 0$ (23) is independent of (24) and for systems with $\frac{\partial g}{\partial x} \equiv 0$ (24) is independent of (23).

†This special case has been examined in detail in earlier work. See [1], [2], [3].

Note that when the latter condition applies $\alpha(t)$ is neglibibly small if $p \geq 2$. Note also that (23) and (24) being uncoupled does not necessarily make either $\gamma(t)$ or $\alpha(t)$ equal to zero.

Equations with Additive Forcing Functions

In the equations (2) and (3) specified for solution on the hybrid computer both g and f can contain implicitly the effect of forcing functions. For equations arising in many practical applications, the forcing functions have an additive property such that f and g can be expressed as

$$f = \hat{f}\{x,z\} + v_1(t) \tag{29}$$

$$g = \hat{g}\{x,z\} + v_2(t) \tag{30}$$

where $v_1(t)$ and $v_2(t)$ are vector forcing functions which account for all explicit variations of f and g with t.

Equations (29) and (30) can be used with any of the forms of the error equation. A tractable equation, for example, results from the use of (29) and (30) in (25), namely

$$\dot{\Gamma}(t) = \frac{\partial h}{\partial u}\dot{\Gamma}(t) + \left[\begin{array}{c} \delta/2\,\dfrac{\partial \hat{f}_a}{\partial z}(\hat{g}+v_2) + \dfrac{\delta+2e}{2}\left[\dfrac{\partial \hat{f}_d}{\partial z}(\hat{g}+v_2) + \dfrac{\partial \hat{f}_d}{\partial x}(\hat{f}+v_1) + V_1(t)\right] \\ - \delta P_\varphi \end{array}\right] \tag{31}$$

where

$$V_1(t) = \begin{cases} \dot{v}_1(t), & v_1 \text{ introduced in the digital computer} \\ 0, & v_1 \text{ introduced in the analog computer} \end{cases}$$

Linear Constant Coefficient Equations

For the hybrid solution of linear constant coefficient equations the error equation reduces to a very tractable form which will be discussed in detail in a later report.

EXAMPLE[†]

To illustrate the use of the error equations in a general but tractable context, consider using a hybrid computer to solve Duffin's equations in the form

$$\dot{u}_1(t) = u_2(t) \tag{32}$$

$$\dot{u}_2(t) = - u_1(t) - .06\, u_1^{\,3}(t) - 2u_2(t) \tag{33}$$

$$u_1(t{=}0) = 4, \; u_2(t{=}0) = 0 \; .$$

Assume that the digital computer uses Euler's method to solve (32), and that (33) is solved on the analog computer with the term $-.06\, u_1^{\,3}(t)$ generated digitally. The general hybrid computer equations then become

$$\dot{x}_1(t) = f_{a1} + f_{d1} = [-2\, x_1(t) - z_1(t)] + [-.06\, z_1^{\,3}(t)] \tag{34}$$

$$\dot{z}_1(t) = x_1(t) \tag{35}$$

$$z_1(t{=}0) = 4, \; x_1(t{=}0) = 0 \; .$$

Use of (23) and (24) yield for the approximate error equations

$$\dot{\gamma}_1(t) = -2\gamma_1(t) - [1 + .18\, z_1^2(t)]\, \alpha_1(t) - \delta/2\, x_1(t) - .18\left(\frac{\delta+2e}{2}\right) z_1^2(t)\, x_1(t) \tag{36}$$

$$\dot{\alpha}_1(t) = \gamma_1(t) - \delta/2[z_1(t) + .06\, z_1^3(t) + 2\, x_1(t)] \tag{37}$$

[†]Several examples illustrating the application of the specialized error equation (27) have been reported earlier. See [3].

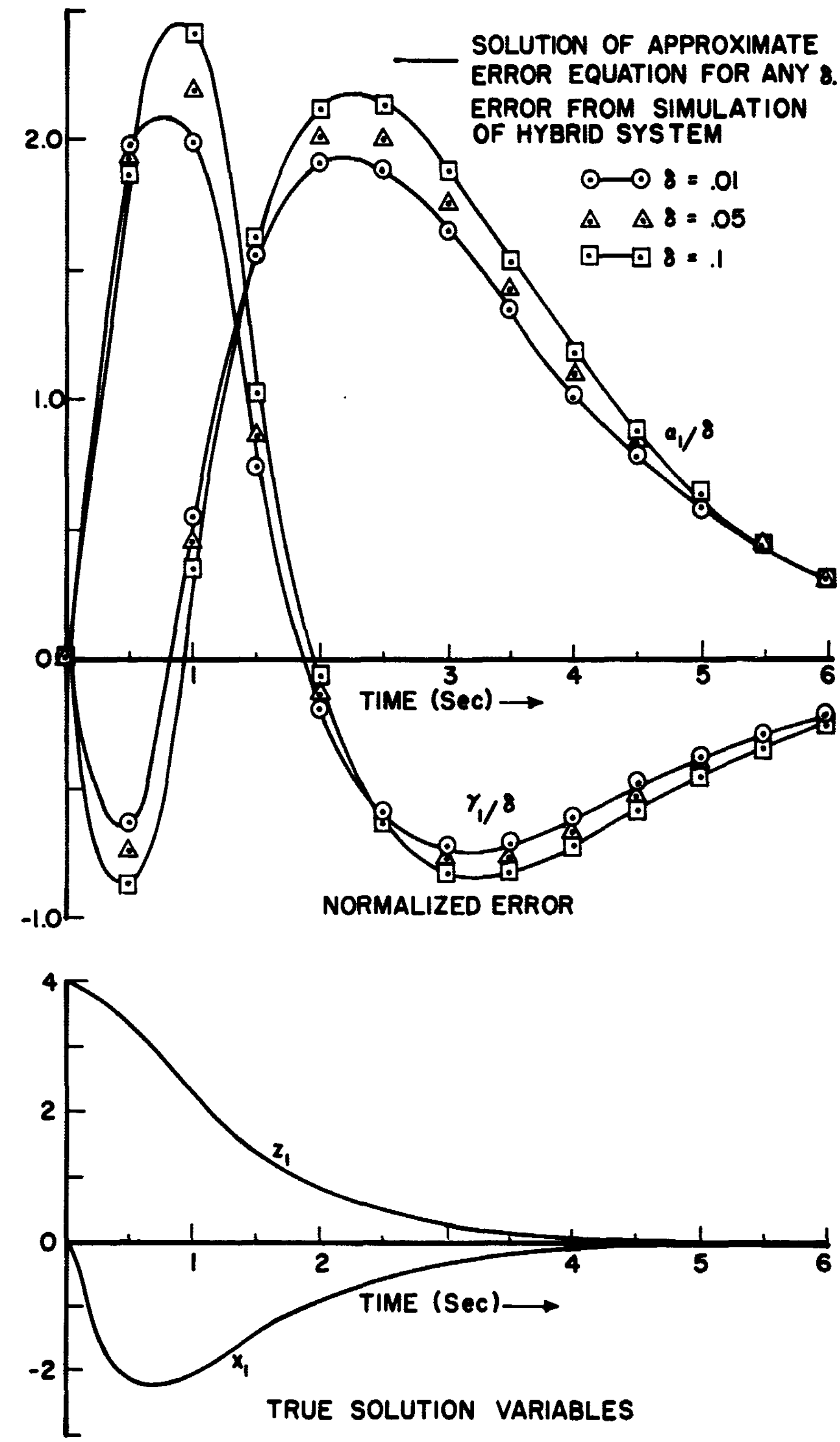

Figure 2. Various Errors and the True Solution Variables for a Hybrid Computer Solution of Duffin's Equation.

In obtaining (37), $\varphi(t)$ (which equals $-\dot{z}(t)/2$ for Euler's method[†]) is expressed by the quantity in square brackets using (35) and (34).

The results of solving (36) and (37) on a digital computer assuming $e = \delta/2$ are shown in Figure 2. The variables α/δ and γ/δ are plotted versus time in the figure and since (36) and (37) are linear the curves apply for any δ. For comparison the errors were also determined by simulating the hybrid computer on a digital computer and subtracting the simulated hybrid variables from the ideal solution variables. Curves of α/δ and γ/δ determined in this manner are shown in the figure for three values of δ. The ideal solution variables $x_1(t)$ and $z_1(t)$ are also given.

Note that the agreement between error computed from (23) and (24) and that determined from the simulated hybrid system is almost exact for $\delta = .01$, becoming less exact but still reasonably good as δ is increased ten fold to .1.

[†]See for example Henrici.[4]

CONCLUSIONS

Equations (23) and (24) or the more compact (25) are differential equations for the sampling error in a general hybrid computer system programmed to solve a vector equation in the form of (1). The equations have evolved in the course of several years of study on the subject contract and include all error equations derived earlier as special cases.

The allocation of operations between the analog and digital computers can be arbitrary but operation of the hybrid computer must conform to the assumptions listed on pages 5 and 6. The error equations hold for a small sampling period and thus could be termed "asymptotic" error equations.

The complexity of the equation is comparable tc that of the given problem equation. However, the equation is an explicit expression for sampling error in terms of computable quantities and thus machine solution is quite feasible. The assumptions as to the form of the problem equation and as to the details of the hybrid implementation, while somewhat restrictive, do not prevent a range of practical applications.

The sampling error equation is expected to have application in three areas, namely:

The evaluation of sampling error for a given hybrid computer implementation.

The study of various allocations of operations between the analog and digital equipment with a view to minimizing sampling error, and

In compensating for sampling error.

Although the major use of (25) is expected to occur in specific applications for which an explicit numerical solution can be obtained, there are several facts of interest which result from examining the forcing function of this equation in its general form. First it should be noted that since (25) is linear each part of the forcing function produces a distinct effect on the solution of the equation, i.e. on sampling error. There are three parts to the forcing function, namely:

(1) $\delta/2 \frac{\partial f_a}{\partial z} g$, which arises because of the discrete nature of z as produced in the digital computer,

(2) $\frac{\delta+2e}{2} \dot{f}_d$, which arises because the vector function f_d is generated digitally for use in the analog computer, and

(3) $\delta^p \varphi$, which is caused by the discretization error of the numerical method in the digital computer.

Since the initial conditions on the error equation are zero, its solution, the sampling error, has three parts corresponding to the three parts of the forcing function. Parts (1) and (2) of the forcing function depend on δ to the first power and thus the hybrid computer, (with zero order sample and hold devices), must be classified as a first order computational method.

The fact that part (3) of the forcing function depends on δ to the pth power brings up an interesting design point. If p, the order of the numerical method, is 2 or greater, the term δ^p is negligible with respect to the other two. Thus since the execution time of the digital computer, (and hence δ) increases with the order of the numerical method, it would seem inefficient to use a numerical method of order greater than 2. Further, it would seem desirable and possible to allot some computer time to compensating for sampling error and increasing the order of the overall hybrid system. This could be done by concurrently solving the error equation, expressed in terms of the non-ideal variables, and using $x^*(t) + \delta(t)$ and $z^*(t) + \alpha(t)$ as compensated variables.

Finally it should be noted from examining part (2) of the forcing function that execution time, e, has the same effect on error as the sampling period but weighted twice as heavily. Thus every effort should be made to keep execution time at a minimum.

APPENDIX

Differentiation of a Vector with Respect to a Vector: Let x be an n vector, y a q vector and g(x,y) an m vector. Then the symbol $\partial g(x,y)/\partial x$ is defined as

$$\frac{\partial g(x,y)}{\partial x} \equiv \left| \begin{array}{ccc} \frac{\partial g_1(x,y)}{\partial x_1} & \cdots & \frac{\partial g_1(x,y)}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial g_m(x,y)}{\partial x_1} & & \frac{\partial g_m(x,y)}{\partial x_m} \end{array} \right| . \qquad \text{(A-1)}$$

Approximation (1): Consider the vector function $g(x_1, x_2+\Delta)$ where Δ is a vector of increments in a portion, x_2, of the vector argument of g. If g is analytic, use of its Taylor series expansion yields the approximation

$$g(x_1, x_2+\Delta) \simeq g(x_1, x_2) + \frac{\partial g(x_1, x_2)}{\partial x_2} \Delta \qquad \text{(A-2)}$$

which neglects terms which are $O(\Delta_i^2)$.

Approximation (2): Consider an analytic function g(t) depending on the scalar argument t. Denote by I_k the intervals $[k\delta, (k+1)\delta)$, k = 0,1,2, ... On any interval I_k the Taylor series for g(t) can be used to obtain the approximation

$$g(k\delta) \simeq g(t) - (t-k\delta)\ g(t), \ t \in I_k \qquad \text{(A-3)}$$

which neglects terms $O(\delta^2)$ since on I_k, $|t-k\delta| < \delta$.

The term $(t-k\delta)$, $k = 0,1,2 \ldots$ in (A-3) is a sawtooth function with period δ and thus has a Fourier series representation of the form

$$(t-k\delta),\ t \in I_k = a_o + S(t) \tag{A-4}$$

where

$$S(t) = \sum_{i=1}^{\infty} (a_i \cos \omega_i t + b_i \sin \omega_i t) \tag{A-5}$$

and $a_o = \delta/2$, $\omega_i = 2\pi i/\delta$, $a_i = 0$, $b_i = -\delta/\pi i$. Using (A-5) in (A-3) yields

$$g(k\delta) \simeq g(t) - [\delta/2 + S(t)]\ \dot{g}(t),\ t \in I_k \tag{A-6}$$

An approximation of the form (A-6) can also be obtained for the function $G(t)$ defined by

$$G(t) = g(k\delta);\ t \in I_{k+e} \tag{A-7}$$

where I_{k+e} is the interval $[k\delta+e, (k+1)\delta + e)$. The result is

$$G(t) \simeq g(t) = [\frac{\delta+2e}{2} - S(t-e)]\ \dot{g}(t) \tag{A-8}$$

and the neglected terms are still $O(\delta^2)$.

Approximation (3): Consider a linear differential equation with the term $S(t)\dot{g}(t)$ from (A-6) as a forcing function. Such an equation can be expressed as

$$\dot{x}(t) = a(t)x(t) + S(t)\dot{g}(t), \tag{A-9}$$

with zero initial conditions. The solution of (A-9) is given by

$$x(t) = \int_0^t \varphi(t;\lambda)\ S(\lambda)\ \dot{g}(\lambda)\ d\lambda\ . \tag{A-10}$$

Under the assumption that the functions $\varphi(t,\lambda)$ and $\dot{g}(\lambda)$ in (A-10) and also their derivatives are bounded and continuous, it can be shown that the solution of the differential equation, $x(t)$, is $0(\delta^2)$.

BIBLIOGRAPHY

[1] Hammond, J. L., "A Study of Error Approximation for Hybrid Computers," Technical Note No. 8, 7 December 1965, (Contract NAS8-2473 Research Project No. A588 Georgia Institute of Technology).

[2] Hammond, J. L., "Approximation for Hybrid Computer Error Due to Sampling Interval and Execution Time," Technical Note No. 12, 30 August 1966, (Contract NAS8-2473 Research Project No. A588 Georgia Institute of Technology).

[3] Hammond, J. L., "Implementation and Evaluation of Equations for Hybrid Computer Error," Technical Note No. 13, 15 January 1967, (Contract NAS8-2473 Research Project No. A588 Georgia Institute of Technology).

[4] Henrici, P., Discrete Variable Methods in Ordinary Differential Equations, New York: John Wiley, 1962.

[5] Moore, R. E., Interval Analysis: Englewood Cliffs: Prentice-Hall Inc., 1966, Chapt. 11.

[6] Reiter, A., "Automatic Generation of Taylor Coefficients (Taylor)" MRC Program #3, COOP Organ, Code-WISC, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin.

On the Generation of a Gaussian Random Process in a Position Parameter