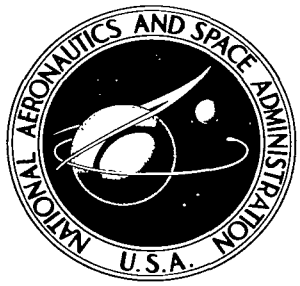
NASA TECHNICAL NOTE

NASA TN D-4203



NASA TN D-4203

PRACTICAL STABILITY CRITERION AND ITS APPLICATION TO DIGITAL SIMULATION

by Leslie L. Scalzott and Carl F. Lorenzo

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • DECEMBER 1967

NASA TN D-4203

PRACTICAL STABILITY CRITERION AND ITS APPLICATION TO DIGITAL SIMULATION

By Leslie L. Scalzott and Carl F. Lorenzo

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

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

PRACTICAL STABILITY CRITERION AND ITS APPLICATION TO DIGITAL SIMULATION

by Leslie L. Scalzott and Carl F. Lorenzo

Lewis Research Center

SUMMARY

A stability criterion has been developed for use with digital simulations or graphical data. The criterion used is a simple one, which basically requires ultimate state boundedness of the time function representing the system output. Techniques are established so that a transient of finite duration may be used as a practical indication of stability.

A stability test utilizing the criterion has been implemented into a general digital program which will allow the determination of system stability directly by computer. This program and criterion was demonstrated for three applications representing: (1) a linear system, (2) a continuously nonlinear system, and (3) a discontinuously nonlinear system.

INTRODUCTION

Currently, there is an increased interest in simulation of physical systems using digital computers. This is especially true of highly nonlinear systems and systems of great complexity both linear and nonlinear. Of particular interest in such systems is the response of the system to either a change in operating point or response to some form of input disturbance. Generally it is desirable to form a stability map showing the effect of two or more system parameters on stability, where stability is some desirable property defined by the investigator.

Presently, to obtain a stability map of a simulated system, a great many transients must be simulated in some organized method, all these transients must be examined to determine whether or not the response is stable by some criterion, and either these results must be mapped directly or further refined with more transient responses. Clearly it would be desirable to eliminate reading all these transient responses out of the digital computer to form the stability map. Indeed, if a computer program could be evolved

which would indicate the stability of the system, the user would be saved by the burden of applying the stability criteria to the transients and the time he must wait between running separate transients. Thus a stability map could be found directly, without the investigator in the loop.

A substantial background of material exists in the literature in the general area of stability. References 1 and 2 are examples of papers that deal with the general problem. Most of the work had as its objective the analytical determination of stability for systems of differential equations. Notable, of course, are the classical efforts of Liapunov, Poincaré, and Lagrange. Current efforts in the English literature (much of the effort has been by Russian investigators) are those of Bellman (ref. 3) and LaSalle and Lefshetz (ref. 4).

The above efforts, as stated, are directed toward analytical solutions of the stability problem. To the authors' knowledge, no work has been done in implementing a stability criterion that can be applied directly to the numerical solutions of systems of differential equations. Direct implementation of these classical criteria is not practical, since it would require an unperturbed solution and the establishment of the classical $\epsilon - \delta$ relation (necessitating a large number of ϵ trials). (Symbols are defined in appendix A.) Reference 2 presents a discussion of this classical criteria. Instead, the present study requires the solutions to have certain desirable properties which approximate the character of those classical criteria. These required properties are then formalized into a working stability criterion. Such a stability criterion and its implementation into a digital program are the subjects of this report.

The results of this study should also be useful in the areas of digital adaptive control systems, data analysis, and as a working criterion for use with experimental studies.

GENERAL SCHEME FOR STABILITY ANALYSIS

The general scheme for determining the stability of a time function generated by digital simulation can be understood by reference to figure 1. There are three basic elements under consideration: (1) the simulation proper, which generates an output time

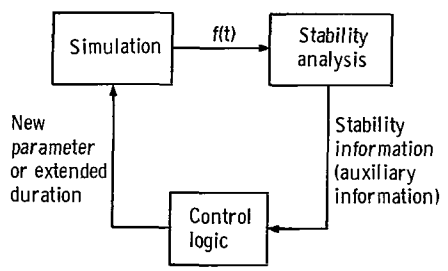


Figure 1. - Block diagram for computer stability analysis.

function in serial form, (2) a routine, which examines the time function and determines its stability, and (3) a control logic routine, which would change the system parameters to achieve the neutral stability points.

Since a practical approximation to stability is being developed, it is clear that a transient of infinite duration cannot be examined. Hence, the stability analysis routine must be capable of making a judgment as to system stability based on a finite observation time. The overall scheme, therefore, will be to assign an observation time over which the time function will be analyzed and determined to be either stable, unstable, or neutrally stable. If the observation time is not long enough, the time function will be conditionally stable or conditionally unstable. If a conditional situation develops in the search for a stability map, the control logic also has the function of progressively increasing the simulation time by a time increment until an unconditional stability or instability occurs. The criteria on which the above conditions are based will be defined formally in the next section.

This report basically deals with the stability analysis block. Throughout the studies presented herein, relatively simple control logic is used. The part of the control logic which determines the new system parameters can be complex. This problem is strongly analagous to the problem of determining roots of equations using digital computer techniques, for which an excellent background of material is available (e.g., ref. 5).

The overall problem can be reduced to the following: Given a function of time which represents the output of a system to some bounded input disturbance, determine the system stability in some finite observation time.

Clearly the choice of a value for observation time is critical in that the user must recognize for his system the dynamic elements with the longest response time and choose the observation time appropriately.

In the preceding discussion only a single system output has been examined. If the possibility of separate parts of a simulation independently being stable or unstable exists, then all these outputs must be monitored for stability.

STABILITY CRITERION

The previous section discussed the general problem of digital determination of the stability of a time function. It was premised on the existence of a technique to discern stability using only the perturbed transient responses. The stability analysis block (fig. 1) should be capable of handling any type of time function in order to be generally useful; that is, time functions should include responses which are continuous or discontinuous, oscillatory or nonoscillatory, and periodic or aperiodic. Furthermore, the system responses of both linear and nonlinear systems are of interest. The analysis should be flexible in order to have general applicability and should also be easily implemented

on the digital computer for use with digital simulations and should use computing time economically.

Fundamentally the criterion is an approximation to asymptotic stability; that is, the requirement will be ultimate state boundedness of the output time response.

Stability criterion for infinite observation time. - A system is considered to be stable if, after the completion of a bounded perturbation on the system, its motion never exceeds some ϵ distance from the bias after some time t_1; that is,

$$|\text{Bias} - f(t)| \leq \epsilon \qquad \text{for } t > t_1$$

where

$$\text{Bias} \equiv \lim_{T \to \infty} \frac{1}{T} \int_0^T f(t)dt$$

If there does not exist a time t_1, such that the motion never exceeds the prescribed ϵ distance from the bias, the system is called unstable.

Physically, the stability definition requires that, after the system transients (due to either a perturbation or change in operating point) die out, the system response remains within a prescribed bound about the new operating point. It further considers simple divergences and growing envelopes to be unstable.

If the stability of linear constant coefficient systems based on the roots of the characteristic equation is considered, the above definition of stability is compatible with linear stability for all points with the possible exception of those on the imaginary axis; that is, a neutrally stable linear system may oscillate forever at some amplitude greater than the prescribed ϵ bound and, based on the given definition, would be unstable. This will not be a problem, however, since, in the study of such systems, the neutral stability point (linear stability) will generally be found by approaching from the positive and negative damping sides and will, therefore, yield the same stability map.

The following is a practical criterion based on the preceding infinite observation time definition.

Stability criterion for finite observation time. - A system is considered to be unconditionally stable if, after the completion of a bounded perturbation on the system, the motion never exceeds some ϵ distance from the bias after some time t_1 but before completion of some observation time t_c. It is further required that the slope of the stability curve be negative. The system is conditionally stable if,

under the preceding conditions, the slope of the stability curve is positive. If there does not exist a time $t_1 < t_c$ such that the motion does not exceed the prescribed ϵ distance and if the slope of the stability curve is positive the system is unconditionally unstable. If, under these conditions the stability curve has a negative slope, the system is conditionally unstable. The stability curve is defined as the locus of $|\text{Bias - Extrema}|$.

This criterion is also stated in table I. The difference between the finite and infinite observation time cases is the additional slope requirement for the stability curve. This slope requirement is used as an indication of the future convergence of the envelope of the time function.

TABLE I. - THEORETICAL STABILITY CRITERIA FOR FINITE OBSERVATION TIME

Functions	Condition	Stability curve boundedness requirement	Slope requirement
Oscillatory	Stable, unconditionally	$\|\text{Bias - Extrema}\| \leq \epsilon$	Slope < 0
	Stable, conditionally	$\|\text{Bias - Extrema}\| \leq \epsilon$	Slope > 0
	Unstable, unconditionally	$\|\text{Bias - Extrema}\| > \epsilon$	Slope > 0
	Unstable, conditionally	$\|\text{Bias - Extrema}\| > \epsilon$	Slope < 0
	Neutrally stable	$\|\text{Bias - Extrema}\| \leq \epsilon$	Slope $= 0$
	Neutrally unstable	$\|\text{Bias - Extrema}\| > \epsilon$	Slope $= 0$
Nonoscillatory	Stable, unconditionally	Bias converges	--------
	Unstable, unconditionally	Bias does not converge	--------

From a practical point of view, this definition still presents some difficulties. Consider, for example, a case of a time function with beats. While the function may be bounded, it may be beating very slowly, and give an apparent positive slope for the stability curve. This difficulty, however, can be handled by programming techniques which consider the extrema of the envelope instead of the extrema of the time function. This will be described in detail in the next section.

In essence this finite observation time criterion, modified by the above technique, is the stability analysis which has been implemented in this study. Further checks and features that have been built into the program are discussed in the next section.

This criterion is different from the classical criteria of Liapunov, Poincaré, and Lagrange as discussed in reference 2, since a different goal is desired. The classical cases generally compare the perturbed and unperturbed responses. Also this is done analytically instead of numerically. In the present situation the stability is determined

without necessarily having the mathematical form of the response and without having the unperturbed motion of the system.

DIGITAL PROGRAM

A digital computer program was written based on the criterion presented in the previous section. This program is called DIGSTA (digital stability analysis). A flow diagram of the computer program and a complete FORTRAN IV listing are given in appendix B. All the checks and definitions applied to the system under investigation are illustrated in this appendix. DIGSTA is written in FORTRAN IV computer language which can be readily translated into other languages. A simplified flow diagram of DIGSTA is presented in figure 2 to assist in understanding the general logic of the stability routine. This figure also illustrates the general scheme used for the determination of stability. Here, the following logic pattern is used.

The time function generated by the system response is used as an input f(t). A check is made to see if the bias value has been found. If the bias is not found, a time check is made. If $t < t_c$, the investigation continues with a new f(t) point. If the bias does not converge for $t \geq t_c$, the response cannot be bounded and is therefore considered unstable (see proof in appendix C). This assumes that t_c is chosen to be a large enough finite time to be representative of an infinite time for the system being studied. If the

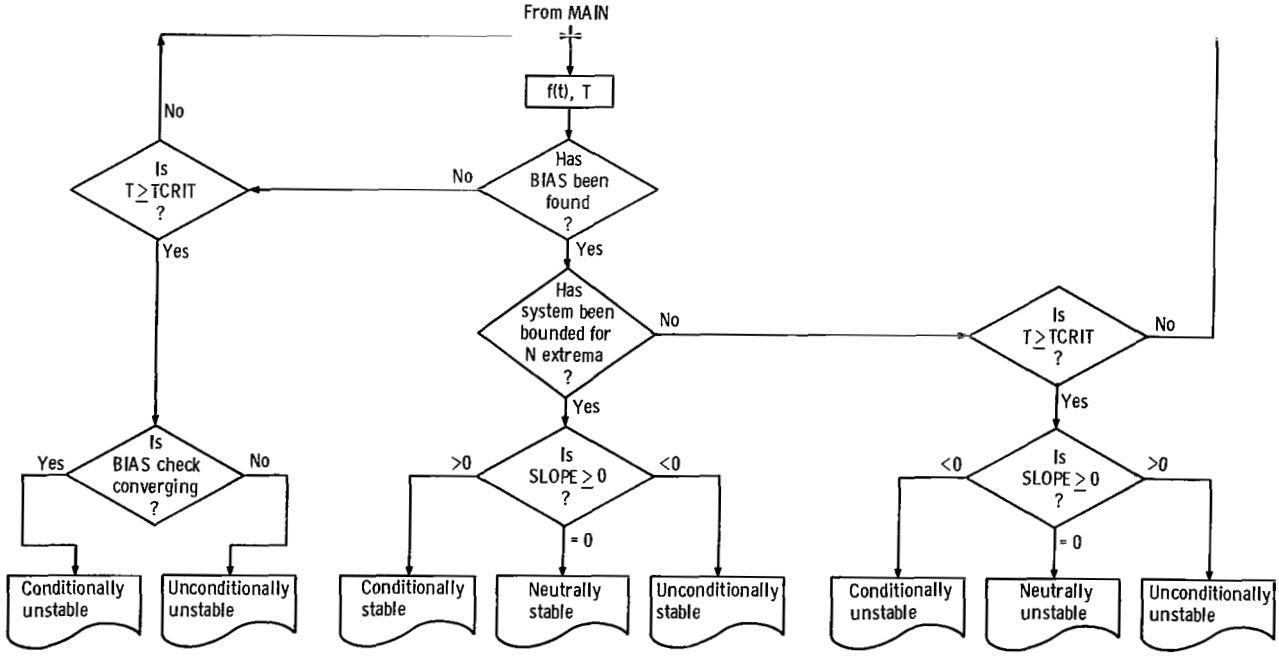


Figure 2. - Simplified flow diagram of DIGSTA.

bias is found, a check is made on boundedness for some prescribed number of extrema. If the boundedness condition is satisfied, the system is stable and the slope of the stability curve is then checked. If the slope is negative, the system is unconditionally stable; if it is positive, it is conditionally stable; and if it is zero, it is called neutrally stable. If the system is not bounded for N extrema in one observation time, a slope check is made again. Herein, positive slopes indicate unconditional instability and negative slopes conditional instability.

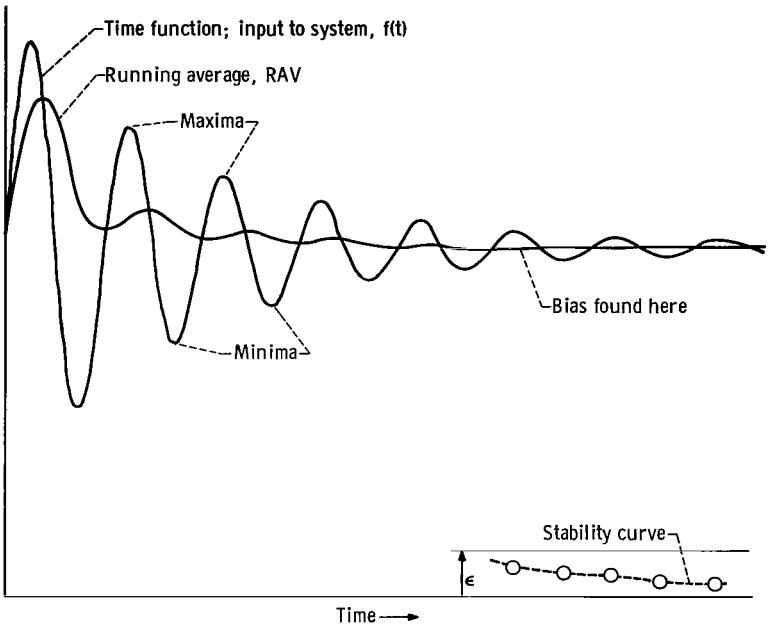


Figure 3. - Typical transient.

There are several terms used in DIGSTA which should be defined or mentioned here. Figure 3 shows a basic curve which could be operated on by DIGSTA. This figure identifies names referred to in this report and in the routine.

Before a boundedness check can be applied, the bias of the system must be established. The bias level has been attained when

$$\frac{d}{dT}\left[\frac{1}{T}\int_0^T f(t)dt\right] = 0$$

In the program the running average (RAV) is defined as

$$RAV \equiv \frac{1}{n\,\Delta t}\sum_{i=1}^{n} f_i(t)\Delta t \qquad n = \frac{t}{\Delta t}$$

The bias is the converged value of the running average and, therefore, ideally requires

$$\frac{d}{dT}(RAV) = 0$$

Two possible numerical requirements for convergence are

$$|RAV_i - RAV_{i-1}| \leq |RAV_{i-1}|\Delta tC$$

and

$$|RAV_i - RAV_{i-1}| \leq \Delta tC$$

The first condition is satisfactory when the RAV is a large number since it requires the running average to converge to within some percentage of the required value. When the bias is near zero, however, RAV is approaching zero. Therefore, the condition requires a smaller and smaller percentage and convergence is not achieved. The second condition is satisfactory around zero; however, large values of RAV could require too small a percentage error. The program, therefore, uses condition 1 for

$$RAV \geq 1$$

and condition 2 for

$$RAV < 1$$

As mentioned previously, the slope of the stability curve is used in defining the system as being conditionally, unconditionally, or neutrally stable or unstable. In DIGSTA only the sign of the slope is of interest and this sign is obtained by an analysis of the signs of each side of the envelope of the function. The sign of one side is defined as the arithmetic sign of $(EXTR_i - EXTR_{i-2})$ and is called SLOPE1. The sign of the other side is defined as the arithmetic sign of $(EXTR_{i-1} - EXTR_{i-3})$ and is called SLOPE2. If sign of $(EXTR_i - EXTR_{i-1})$ is positive (negative) then SLOPE1 is the upper envelope of the curve and SLOPE2 is the lower envelope and conversely if $(EXTR_i - EXTR_{i-1})$ is negative. The sign of the slope of the stability curve then is given by $SGN(MM\ SGN(EXTR_i - EXTR_{i-1}))$ where the dummy variable MM is determined from table II.

TABLE II. - DUMMY VARIABLE AS FUNCTION OF SLOPE

Slope 2	Slope 1		
	+	-	0
+	±9.9	-1.0	-1.0
-	+1.0	±9.9	+1.0
0	+1.0	-1.0	0

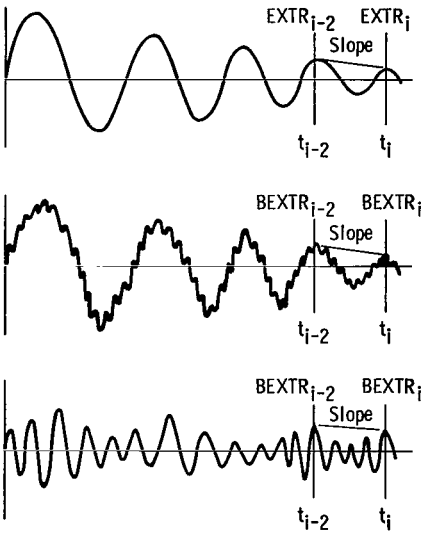


Figure 4. - Beat and high-frequency phenomena.

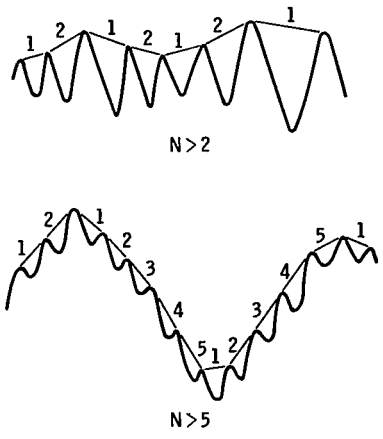


Figure 5. - Choice of index number.

Now, a -1.0 value of MM SGN($EXTR_i$ - $EXTR_{i-1}$) indicates that the envelope is converging and a 1.0 value indicates divergence. Then this product together with the boundedness requirement gives a stability indication.

The +9.9 value denotes that the signs of the envelopes are contradictory (i.e., both positive or both negative) and is a warning to the investigator. Herein, +9.9 indicates an unbounded function and -9.9 indicates bounded.

DIGSTA performs a beat and high frequency oscillation investigations on the function of interest. This is done by examining the envelope of each side of the time function for changes in the arithmetic sign of the slope. Changes in sign are an indication of those phenomena. When this occurs, DIGSTA refers to the extrema of the extrema (BEXTR) in lieu of EXTR. This method is illustrated graphically in figure 4.

Since DIGSTA is a routine used in conjunction with a main program, certain constants and variables must be made common to both programs. These terms are IPRINT, K1K2, L, M, N, DELTAT, EPSLON, T, TPREV, Y, YP, TCRIT, and TINCR. They are defined in appendix A and values are recommended where applicable.

Certain discretion must be exercised in defining some of the constants. In particular, N must be chosen so as to be sufficiently large to account for peculiarities encountered when investigating a system that is beating or has high-frequency oscillations superimposed; that is, N must be larger than the number of consecutive, similarly signed slopes of odd-numbered intervening extrema (see fig. 5).

In addition to being the number of successive stability curve points required to be within an ϵ-band of the bias for a system to be defined as being stable, N is also used as

(1) The number of successive slopes required to define a beat or high frequency oscillations

(2) The number of successive slopes required to define a system as no longer beating or having high frequency oscillations (2N)

If the user desires to alter any of these criteria, he can readily do so by adjusting the pertinent FORTRAN "IF statement" in DIGSTA.

Similarly, the user must have an insight into his problem so he can select reasonable TCRIT, TINCR, DELTAT, and EPSLON values. Adequate time must be allowed to permit the program to solve for the bias and then apply the system stability investigation.

In most applications the settling time can generally be gaged by examination of those system components with the longest characteristic time. For first order systems the time constant τ is the controlling factor, and for the second order, the parameter $1/\zeta\omega_N$ is the controlling factor. A choice of TCRIT of 10 or more times the greatest characteristic time should be suitable for most "linear like" systems. Smaller ϵ values require larger TCRIT values.

For an unstable configuration this amount of time should be adequate to establish the instability. The critical factor here is that the user at least allows adequate time (TCRIT) to solve for the bias for the stable situation.

It is the authors' feeling that the BIAS as presently defined is satisfactory for most purposes although somewhat slow to converge. If, after TCRIT has elapsed, the system has been identified as being conditionally stable or unstable, the user may deem it appropriate to increase the observation time. Therefore, TINCR may be assigned a nonzero value. Also, DELTAT (Δt) must be chosen to be sufficiently small to reasonably define the extrema. Using information concerning the system under investigation, the user defines an EPSLON value in the main program which establishes the maximum acceptable amplitude for system stability.

One merit of DIGSTA is the small amount of storage space required (113 storage locations). Also, as a typical example, the relay controller simulation and its logic section required 64 storage locations. This total is small relative to that available on the larger computers. This advantage will prove particularly useful when DIGSTA is coupled with a main program that requires a large number of storage locations. Also, DIGSTA can be easily implemented to the program of interest since either equations or data can be operated on by the stability program.

To handle physical data containing noise, some care is necessary in the selection of the N number as discussed previously. Another asset of DIGSTA is the small additional time requirement. An exponentially decaying cosine function was operated on by DIGSTA and was found to be stable after 0.19 minute of computer time and 18.101 seconds of simulation time (DELTAT = 0.001 sec). This transient was then run without DIGSTA for 18.101 seconds of simulation time. The computer time required for this case was

0.09 minute. This example should not be interpreted to mean that the use of DIGSTA doubles the time required. The transient routine could be quite complex, and it could have a large computer time requirement. This example merely demonstrates that 18 101 data points were analyzed by DIGSTA in 0.10 minute. This may be used to estimate the time required by DIGSTA.

Control Logic

Some control logic is needed to adjust the system parameter being varied to achieve the stability locus. In the applications that follow an elementary technique was used to iterate to the neutrally stable condition. Two values were chosen to be the bounds of the parameter. One of these was assumed to be stable and the other unstable. An initial parameter value between the two limits was chosen. If the system was then found to be stable, a constant value was added to the initial guess until a parameter value was found that rendered an unstable system (and, conversely, for the unstable system). Then it was necessary to iterate between these last two values. The method used was that of choosing a value midway between these extremities and thereby establishing new bounds. This process was repeated until the two limits were an assigned percentage apart. Then, the neutrally stable point was defined as the stable limit. This technique can be modified or replaced in order to increase efficiency. One alternative for linear-like systems would be to use the slope of the envelope of the last trial as a basis for selecting a new trial point.

For the computer used, there is a restart capability. This means that, in the event that the adjustable parameter is so chosen that there is an underflow ($X \leq 10^{-38}$) or an overflow ($X \geq 10^{38}$) or a division by zero (computer arithmetic errors), the control logic section can then adjust the parameter and the investigation can be continued. DIGSTA has been written to allow reentering the stability subroutine after such a computer operation stoppage without a loss of continuity.

APPLICATION OF DIGITAL PROGRAM

The program developed in the preceding section has been applied to a series of elementary functions to verify program operation and has also been applied to three cases of interest to demonstrate use of the program.

The elementary functions were sines and cosines with exponential decay and growth functions as multipliers. These were studied with and without bias and combinations of functions were studied to simulate a beating situation. All these functions gave proper

stability indications with the criterion as previously defined.

The three practical cases include study of (1) the stability of a linear system, (2) a nonlinear system with a continuous type (mild) nonlinearity, and (3) a system with a discontinuous type (hard) nonlinearity.

Linear System Stability

A stability study was made of the dynamics of a simple monopropellant rocket engine system with a linear injector. It consists of an injector of the form $\dot{w}_i = K(P_T - P_C)$, a pure dead time representing propellant vaporization time and a first order lag representation of the chamber fill dynamics. The equations defining this system are as follows:

$$\tau \frac{dP_C}{dt} + P_C = K_1 \dot{w}_b$$

$$\dot{w}_b(t) = \dot{w}_i(t - \sigma)$$

$$\dot{w}_i = K_2(P_T - P_C) = K_2 \, \Delta P$$

These equations are expressed in block diagram form in figure 6, using Laplace notation.

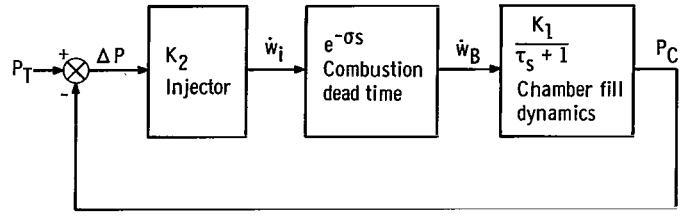


Figure 6. - Block diagram for rocket engine stability study.

Numerical approximations to the above equations were used to generate the time function required for the stability analysis.

The following nominal conditions were used:

Chamber pressure differential, P_C, psi; N/m^2	682; 4.70×10^6
Weight flow, $\dot{w}$, lb/sec; kg/sec	463; 210
Chamber gain constant, K_1, sec/in.^{2}; sec/cm^2	1.4721; 0.2282
Delay time, σ, sec	0.002
Time constant, τ, sec	0.001

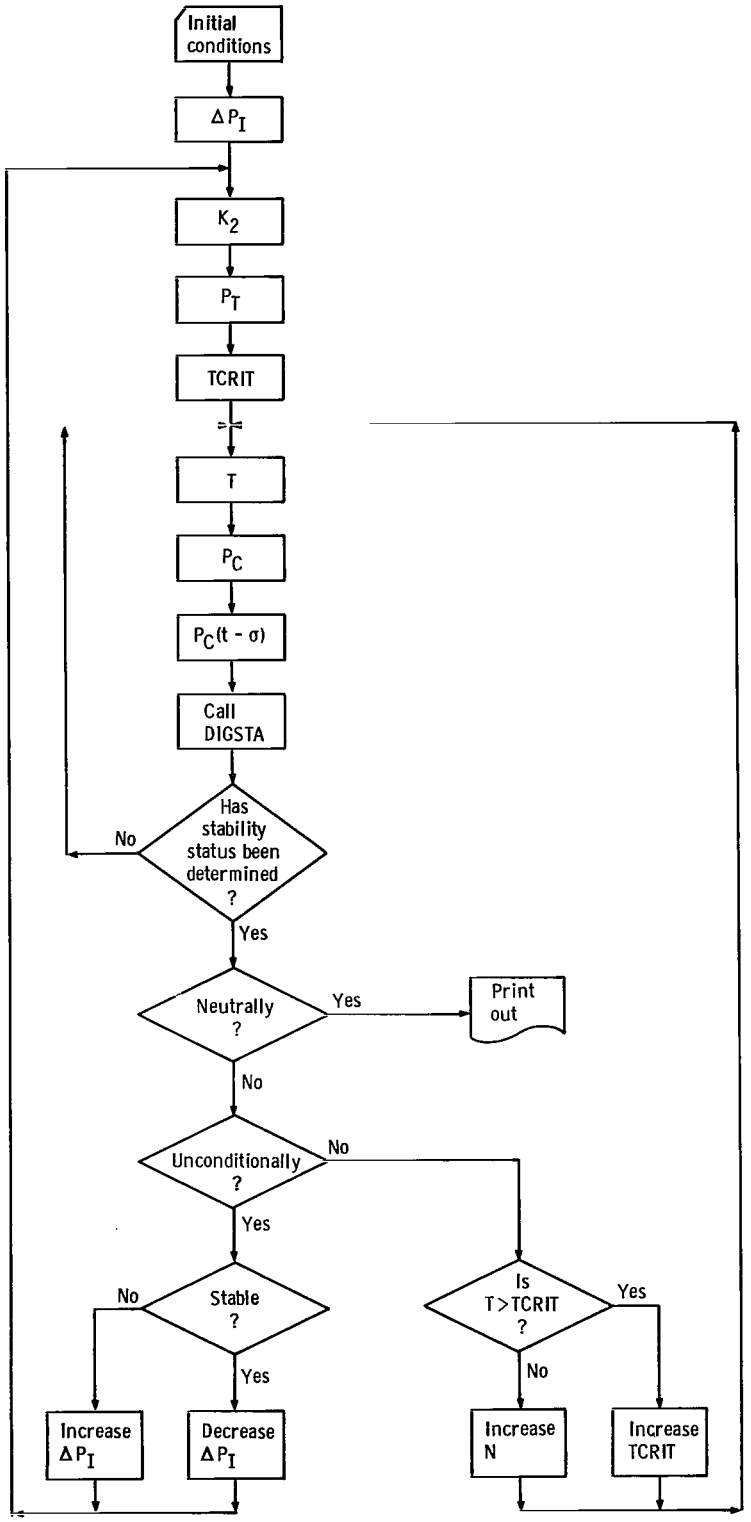


Figure 7. - Digital implementation for rocket engine chugging problem.

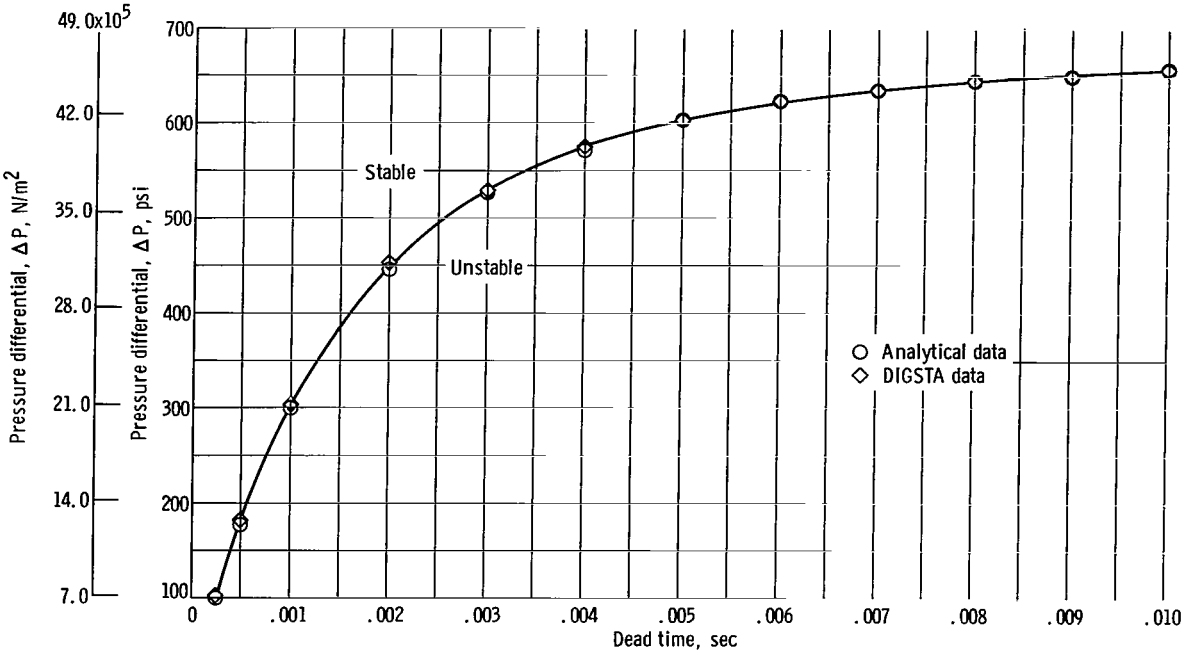


Figure 8. - Stability map for rocket engine with linear injector.

The critical time t_c for this analysis was chosen as 1 second. Also $N = 6$ and a Δt of 0.00001 second were used. A flow diagram which shows how the simulation, stability subroutine and control logic were combined is presented in figure 7. For ϵ = 1500 psi (10.34×10^6 N/m^2), the program was asked to determine the injector ΔP to within 1 psi (6.89×10^3 N/m^2) required for stability for a given dead time. This, in turn, was repeated for the number of dead time values required to form the graph of figure 8. The disturbance used to excite the system was a step in the tank pressure P_T from 0 to 1130 psi (0 to 7.79×10^6 N/m^2) at time zero.

The area above the curve in the figure indicates injector pressure drops for which the system is stable, and that below the curve, unstable. It will be noted that stability as determined from the roots of the characteristic equation (see ref. 6 for technique) fell on the same line within readability limits.

To get the 12 stability points shown in this figure required the simulation of 156 transients. The ϵ value of 1500 psi (10.34×10^6 N/m^2) was used in the stability criterion since the initial large transient would have taken significant time to arrive within a smaller ϵ band. This is permissible since linear system stability is being studied.

Nonlinear System Stability (Continuous Nonlinearity)

In order to study a mild nonlinearity, the rocket system simulation discussed in the above section was modified by assuming a nonlinear injector characteristic; that is,

$$\dot{w}_i = K_2 \sqrt{P_T - P_C}$$

Also, $(P_T - P_C)$ was limited to zero so that the back flows were not possible. The nominal conditions are the same as the above case. For these conditions, ϵ values of 10 and 100 psi (6.89×10^4 to 68.9×10^4 N/m^2) were studied. The stability of this system in response to the same input as studied previously is shown in figure 9.

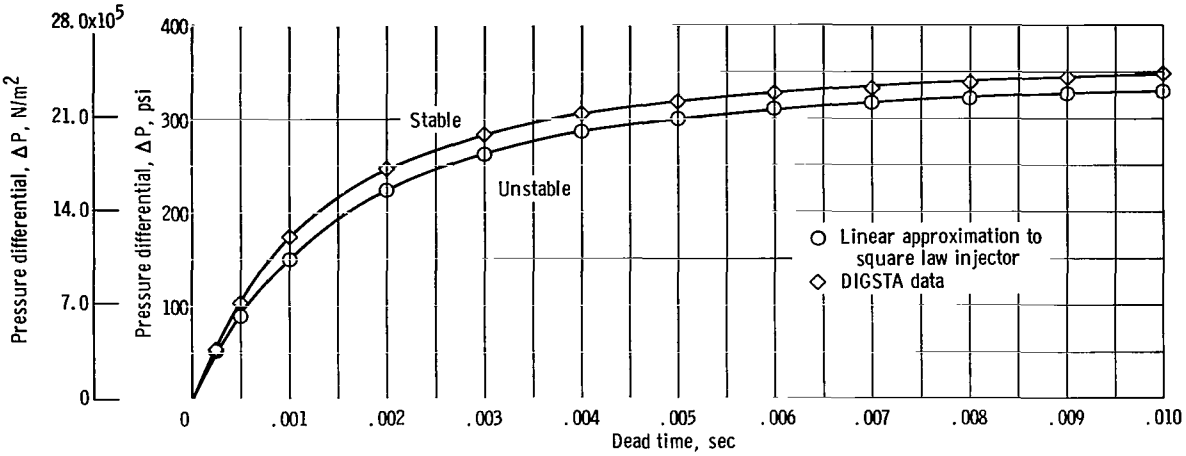


Figure 9. - Stability map for rocket engine with square law injector.

Normally, to handle this type of nonlinearity, the method of small perturbations would be applied to linearize the system equations. These, in turn, are analyzed based on the roots of the characteristic equation. This analysis has been done and the results of such a linearization are also plotted in figure 9. The difference is an indication of the effect of the nonlinearity on system stability. The effect of the square law injector is to distort the weight flow wave shape. This distortion shifts the average-value weight flow during dynamic oscillations, which, in turn, increases the pressure drop required to stabilize the system.

The magnitude of the bias shift depends on wave amplitude. Thus, the magnitude of the difference between the stability curves is dependent on the size or harshness of the input disturbance.

In this study the smaller values entailed longer simulation times than would be encountered in the linear case.

Nonlinear System Stability (Discontinuous Nonlinearity)

For this application, a stability analysis was made for a system in which a relay is used to control the position of an inertial torsional servo. The block diagram for this

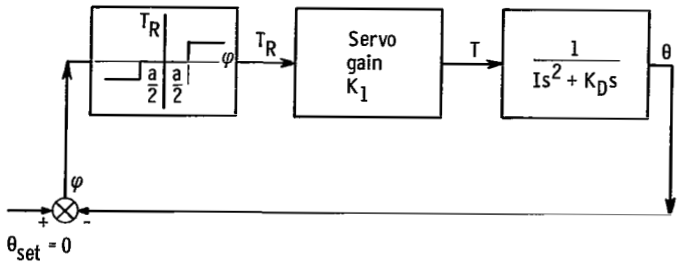


Figure 10. - Block diagram for position control torsional servo.

system is shown in figure 10. The defining equations for this system are

$$T = I\frac{d^2\theta}{dt^2} + K_D\frac{d\theta}{dt}$$

$$T = K_1 T_R$$

$$T_R = \begin{cases} +1 & \frac{a}{2} < \varphi \\ 0 & -\frac{a}{2} \leq \varphi \leq \frac{a}{2} \\ -1 & \varphi < -\frac{a}{2} \end{cases}$$

$$\varphi = \theta_{set} - \theta$$

The relay controller is being studied, to demonstrate the use of DIGSTA for limit cycle forms of operation. The nominal conditions for this analysis were

Damping constant, K_D, ft-lb-sec; m-kg-sec . . . . . 0.0001; 0.138×10^{-4}
Moment of inertia, I, ft-lb-sec; m-kg-sec . . . . . 1.000; 0.138×10^{-4}
Limit of angular position, a, rad 0.01745
Desired θ value, θ_{set}, rad 0

In the DIGSTA subroutine the following values were used:

Critical time, t_c, sec 50
Number of extrema, N 5

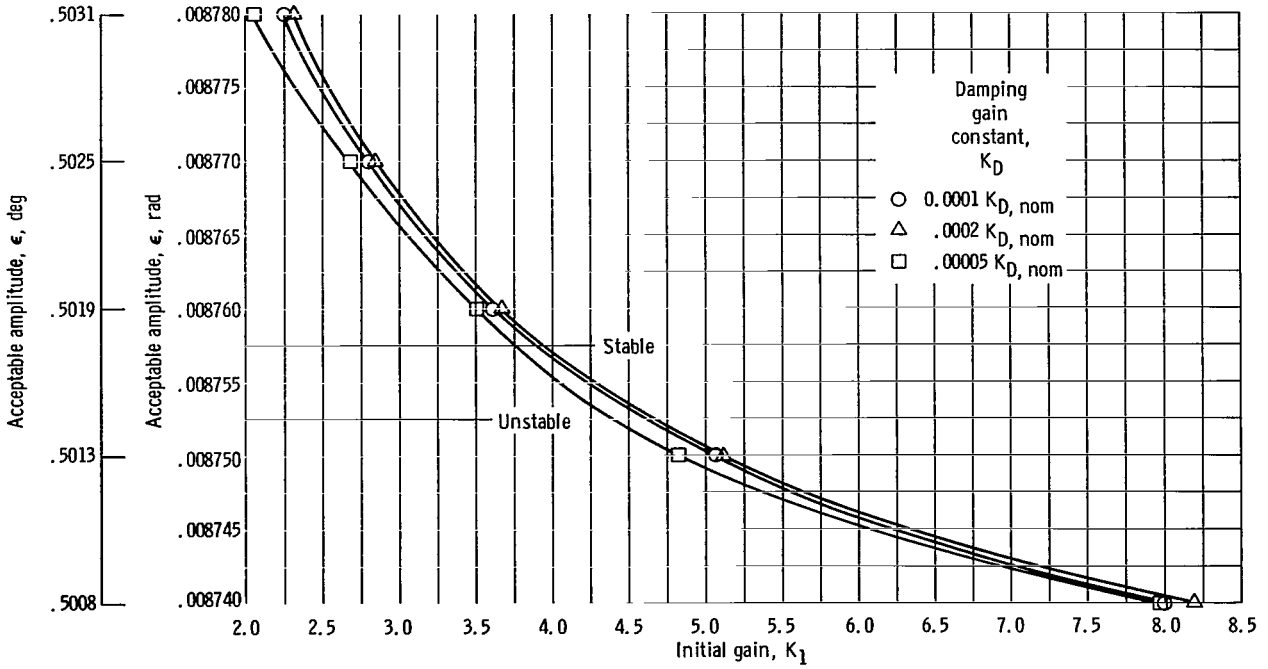


Figure 11. - Stability map for torsional position control servo.

In this particular study, some difficulty was encountered in arriving at a valid simulation. As a result, since the problem is piecewise linear, the differential equations governing the behavior of the output over any time interval were solved analytically. This yielded an algebraic equation which in turn was solved to generate the system outputs. This equation was solved in a serial form, which formed an adequate simulation input for the digital subroutine. The relay dead band was ±0.0087265 radian ($\pm 1/2^{o}$). The disturbance was an initial angular velocity of 0.005 radian per second.

The results of this study are shown in figure 11. Here, for a given value of damping a K_1 (within 0.001) was established which would indicate stability for some value of ϵ. In this case, the ϵ value corresponds to the acceptable limit cycle amplitude. Also indicated in this study are the effects of variations of servodamping K_D.

The study involving limit cycles presented some small difficulty since long simulation times were required to arrive at final stability indications. For $K_D = 1 \times 10^{-4}$ and $\epsilon = 0.00878$ radian, 44.7 seconds of transient were required. This is due to the slow convergence or divergence of the envelope inherent in this problem.

The above are typical applications in linear and nonlinear controls and dynamics systems. The program could also be used with a real system as an indication of stability. This is possible since the time function generated from the computer simulations are in serial form, and the subroutine would not see any difference. Some problems in control logic would exist. However, it might not even be necessary to have a control logic unit if conditional stability indications were acceptable for a given application.

This ability to examine real systems opens the avenue to possible applications in digital adaptive control systems.

SUMMARY OF RESULTS

The most important result of this work is that a stability criterion has been evolved which will allow a stability analysis of systems for bounded, finite duration inputs. The criterion is general in that it will function properly for practical time functions, and it is flexible to the extent that the user can pick his boundedness band and special index numbers. The techniques evolved in the present study can also be used as a basis for different or more sophisticated criteria. The problem of working with a finite observation time has been satisfactorily handled by use of the stability curve, that is, the information in the envelope of the time function.

The stability criterion has been implemented for use with the digital computer and requires modest additional computing time to determine the stability bounds directly, furthermore, a FORTRAN program implementation of the criterion together with a simple control logic has been presented.

The program has been demonstrated for three applications which cover a linear, continuously nonlinear, and discontinuously nonlinear dynamic systems. For these systems stability maps have been generated directly from the computer without the need of examining separate system transients.

The program allows the reduction of turn-around time in the use of the digital computer for mapping stability results, and allows a uniform basis for stability analysis.

The control logic used in the applications presented has been simple, it is likely that a significant economy in computing time can be gained by implementing more sophisticated techniques, that is, techniques analogous to those used for root finding of equations.

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, June 29, 1967,
180-31-01-05-22.

APPENDIX A

SYMBOLS

Engineering Symbols:

a	θ limit, rad
C	convergence constant
I	moment of inertia, lb-ft-sec^2; kg-m-sec^2
K	gain constant
K_1	chamber gain
K_2	injector gain
P	pressure, psi; N/sq m
ΔP	pressure differential, psi; N/sq m
s	Laplace variable
T	torque
t	time
Δt	time increment
t_c	critical time
$\dot{w}$	weight flow, lb/sec; kg/sec
θ	angular position
θ_{set}	desired θ value
φ	position error
ϵ	allowable oscillation amplitude
σ	delay time
τ	time constant
ζ	damping constant
ω_N	natural frequency, rad/sec

Subscripts:

b	burning
C	chamber
D	damping
I	injector
i	current

R	required
T	tank
1	initial

FORTRAN Variables Common to Both the Main Program and the Stability Routine:

DELTAT	time increment; defined in MAIN
EPSLON	allowable oscillation magnitude; defined in MAIN
IPRINT	when IPRINT = 0, there is a printout every stable or unstable transient; if IPRINT ≠ 0, there is a printout only for a neutral point (slope = 0); defined in MAIN
K1K2	identifies the system as being stable, unstable, conditionally, unconditionally or neutrally stable or unstable. Utilized by subroutine change logic; calculated in DIGSTA
L	equal to 1 when f(t) is not equal to constant for first time; set in MAIN
M	equal to 0 until DIGSTA is entered; set in MAIN
N	number of successive stability curve points required to be within ϵ-band of bias; defined in MAIN
T	current system running time; developed in MAIN
TCRIT	allowable time to identify the behavior of the system; defined in MAIN
TINCR	allowable increment to increase TCRIT; defined in MAIN
TPREV	previous T; defined in MAIN
Y	variable being tested in DIGSTA; developed in MAIN f(t)
YP	previous Y; defined in MAIN

Other FORTRAN Variables Used in the Stability Program:

BIAS	converged value of the running average
EXTR	extrema
KOUNT1	number of successive points (BIAS) identical within ϵ_1-band
KOUNT2	number of successive stability curve points within ϵ-band of BIAS
RAV	running average

APPENDIX B

FORTRAN PROGRAM

To assist in explaining the computer coding of the DIGSTA stability analysis, a flow diagram (fig. 12) and a listing of the program are presented.

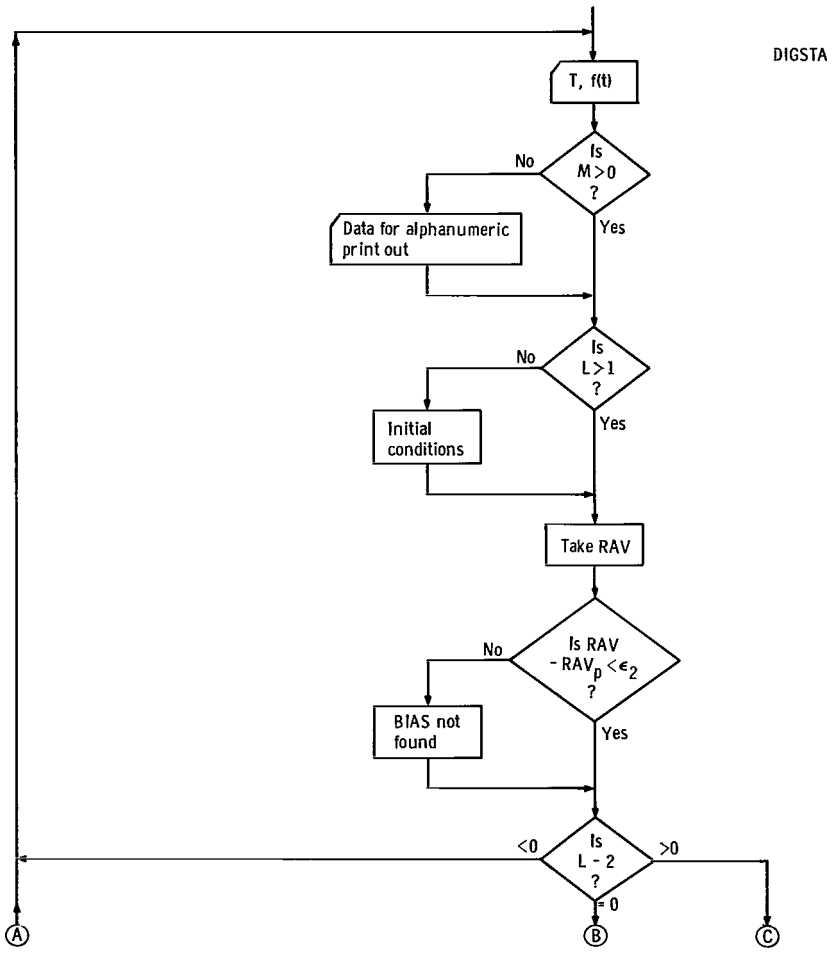


Figure 12. - Flow diagram of computer program.

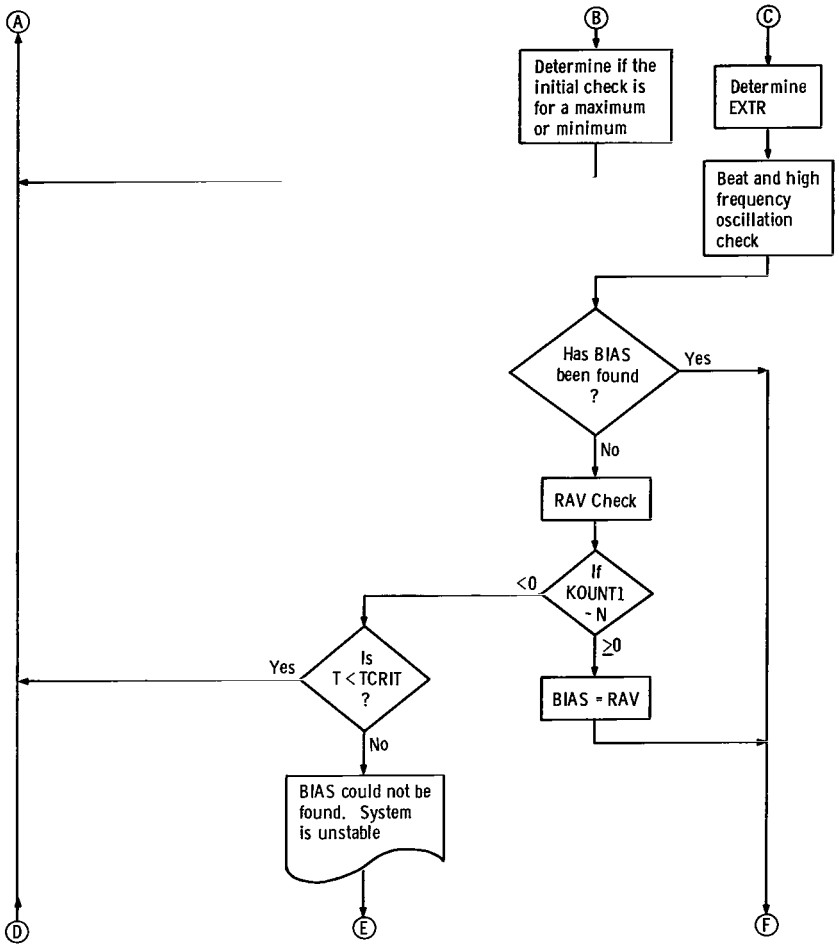


Figure 12. - Continued.

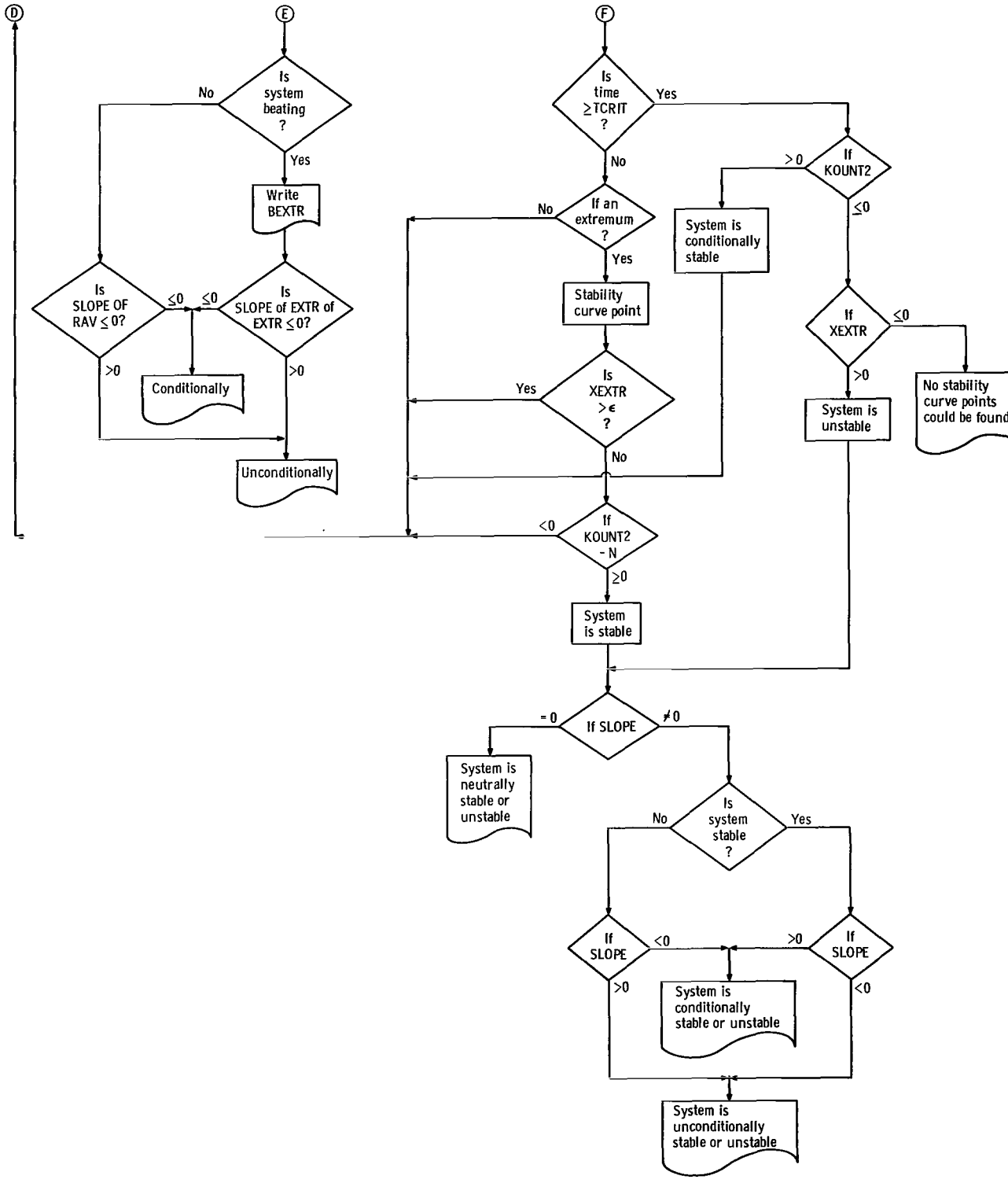


Figure 12. - Concluded.

```
      SUBROUTINE DIGSTA (IPRINT,K1K2,L,M,N,DELTAT,EPSLON,T,TPREV,Y,YP,
     1     TCRIT,TINCR)
C
C     DELTAT, EPSLON, TCRIT, TINCR, IPRINT, AND N MUST BE INITIALIZED IN THE
C     MAIN PROGRAM
C
C     T, TP, Y, YP, L, AND M MUST BE CALCULATED IN THE MAIN PROGRAM
C
      LOGICAL A,B,C,D,E,F
      DIMENSION XA(11)
      IF (M .GT. 0)                                          GO TO 20
      READ  (5,7000) (XA(II),II=1,11)
      WRITE (6,7000) (XA(II),II=1,11)
   20 D      = .TRUE.
      IF (L .GT. 1)                                          GO TO 40
C
C     A       = TRUE  - SEARCHING FOR MINIMIM
C     B       = FALSE - TO RESET KOUNT1 = 0
C     C       = TRUE  - BIAS HAS BEEN FOUND
C     D       = FALSE - EXTREMUM HAS BEEN FOUND
C     E       = TRUE  - NO BEAT PHENOMENA OR HIGH FREQUENCY OSCIL. HAVE OCCURRED
C     F       = TRUE  - NO SECOND ORDER BEAT PHENOMENA OR HIGH FREQUENCY OSC.
C     IPRINT = 0 - INTERMEDIATE PRINT OUTS ELIMINATED
C     J       = NO. OF EXTREMA
C     JJ      = NO. OF EXTREMA OF EXTREMA
C     K       = NO. OF S.C. POINTS
C     KOUNT1 = NO. OF SUCCESSIVE PTS. (BIAS) IDENTICAL WITHIN E1 BAND
C     KOUNT2 = NO. OF SUCCESSIVE S.C. PTS. WITHIN E BAND OF BIAS
C     KOUNT3 = NO. OF SUCCESSIVE SLOPES INDICATING A BEAT OR HIGH FREQ. OSC.
C     KOUNT4 = NO. OF SUCCESSIVE SLOPES INDICATING NO BEAT OR HIGH FREQ. OSC.
C     KOUNT5 = NO. OF SUCCESSIVE SLOPES INDICATING A SECOND ORDER BEAT OR
C              HIGH FREQ. OSC.
C     KOUNT6 = NO. OF SUCCESSIVE SLOPES INDICATING NO SECOND ORDER BEAT OR
C              HIGH FREQ. OSC.
C     K1K2    = K1 + K2, INDICATES NATURE OF STABILITY OF THE SYSTEM
C     L       = 1 WHEN Y IS NOT ZERO FOR THE FIRST TIME
C     M       = 0 WHEN THE SUBROUTINE IS ENTERED FOR THE FIRST TIME
C     N       = NO. OF SUCCESSIVE S.C. PTS. REQUIRED TO BE WITHIN E BAND OF BIAS
C     BEXTR   = EXTREMA OF EXTREMA
C     EXTR    = EXTREMA
C     RAV     = RUNNING AVERAGE OF DATA POINTS FOR THE SOLUTION OF THE BIAS
C     RX      = VALUE OF RUNNING AVERAGE WHEN KOUNT1 = 0
C
C     XEXTR   = STABILITY CURVE (S.C.) POINT (LOCUS OF EXTREMA)
C    INITIAL  CONDITIONS  FOR  RUNNING  AVERAGE  AND  STABILITY  CURVE  CHECKS
C
      BIAS    = 1.0E+35
      BEXTR   =   0.0
      BEXTP   =   0.0
      BEXTPP  =   0.0
      BEXTP3  =   0.0
      CEXTR   =   0.0
      CEXTP   =   0.0
      CEXTPP  =   0.0
      CEXTP3  =   0.0
      EXTR    =   0.0
      EXTRP   =   0.0
      EXTRPP  =   0.0
      RAV     =   0.0
      RX      =   0.0
```

```
      SLOPE1 =    0.0
      SLOPE2 =    0.0
      TB     =    0.0
      TBP    =    0.0
      TBPP   =    0.0
      TC     =    0.0
      TCP    =    0.0
      TCPP   =    0.0
      TT     =    0.0
      TTP    =    0.0
      TTPP   =    0.0
      TRAV   =    0.0
      X      =    0.0
      XEXTR  =    0.0
      XRAV   =    0.0
      XY     =    0.0
      Y1     =        Y
      Z      =    0.0
      IK     =    0
      J      =    0
      JJ     =    0
      JB     =    0
      JBP    =    0
      JS     =    0
      JSP    =    0
      K      =    0
      KI     =    0
      K1K2   =    0
      KOUNT1 =    0
      KOUNT2 =    0
      KOUNT3 =    0
      KOUNT4 =    0
      KOUNT5 =    0
      KOUNT6 =    0
      B      = .FALSE.
      C      = .FALSE.
      E      = .TRUE.
      F      = .TRUE.
C
C                              RUNNING  AVERAGE  CHECK
C
   40 RPP    = RP
   50 RP     = RAV
      IF (T .EQ. 0.0)                                    GO TO 70
      RAV    = (RP*TPREV/DELTAT + Y)*DELTAT/T
      IF (ABS(RAV - RX) .LE. XY)                         GO TO 70
      C      = .FALSE.
      BIAS   =   1.0E+35
      TBIAS  =   0.0
      KOUNT1 =   0
   70 IF (L - 2)                                         1150,   80,   110
   80 Y2     = Y
      IF (Y2 .GT. Y1)                                    GO TO 100
      A      = .TRUE.
      GO TO 1150
  100 A      = .FALSE.
      GO TO 1150
C
C                           STABILITY  CURVE  CHECK
C
```

```
  110 IF (Y - YP)                                          120,  650,  130
  120 IF (A)                                              GO TO 650
      A       = .TRUE.
      GO TO 150
  130 IF (A)                                              GO TO 140
      GO TO 650
  140 A       = .FALSE.
  150 J       = J + 1
      EXTRP3  = EXTRPP
      EXTRPP  = EXTRP
      EXTRP   = EXTR
      EXTR    = YP
C
C                 BEAT  OR  HIGH  FREQUENCY  OSCILLATION  CHECK
C
      EXEX    = EXTR - EXTRPP
      JSPP    = JSP
      JSP     = JS
      IF (ABS(EXEX) .GT. 1.0E-07*EXTR)                    GO TO 170
      JS      = JSPP
      GO TO 180
  170 JS      = SIGN(1.0,EXEX)
  180 IF (J .LT. 4)                                       GO TO 550
      IF (J/2*2 .NE. J)                                   GO TO 250
      IF (E)                                              GO TO 210
      GO TO 650
  210 IF (JS .EQ. JSPP)                                   GO TO 550
      KOUNT4 =   0
      KOUNT3 = KOUNT3 + 1
      IF (KOUNT3 .LT. N)                                  GO TO 550
      IF (IK .LT. 2)                                      J = J+1
      GO TO 300
  250 IF (JS .EQ. JSPP)                                   GO TO 270
      KOUNT4 =   0
      IF (E)                                              GO TO 290
      GO TO 310
  270 IF (E)                                              GO TO 550
      KOUNT4 = KOUNT4 + 1
      IF (KOUNT4 .LT. 2*N)                                GO TO 310
      E       = .TRUE.
      KOUNT2 =   0
      KOUNT3 =   0
      IK     =   0
      GO TO 550
  290 KOUNT3 = KOUNT3 + 1
      IK      = IK + 1
      IF (KOUNT3 .LT. N)                                  GO TO 550
  300 E     = .FALSE.
      JSPP    = JS
      XYZ5    = EXTRPP
  310 IF (JS)                                              320,  650,  330
  320 XYZ5P   = AMAX1(XYZ5,EXTRP)
      XYZ5    = AMIN1(EXTR,EXTRP,EXTRPP,EXTRP3,XYZ5)
      IF (JSPP)                                            650,  650,  340
  330 XYZ5P   = AMIN1(XYZ5,EXTRP)
      XYZ5    = AMAX1(EXTR,EXTRP,EXTRPP,EXTRP3,XYZ5)
      IF (JSPP)                                            340,  650,  650
  340 BEXTP3 = BEXTPP
      BEXTPP = BEXTP
      BEXTP  = BEXTR
```

```
      BEXTR  = XYZ5P
      KOUNT3 = KOUNT3 + 1
      JJ     = JJ + 1
      BXBX   = BEXTR - BEXTPP
      JBPP   = JBP
      JBP    = JB
      IF (ABS(BXBX) .GT. 1.0E-07*BEXTR)                GO TO 360
      JB     = JBPP
      GO TO 370
  360 JB     = SIGN(1.0,BXBX)
  370 IF (JJ .LT. 4)                                   GO TO 540
      IF (JJ/2*2 .NE. JJ)                              GO TO 440
      IF (F)                                           GO TO 400
      GO TO 650
  400 IF (JB .EQ. JBPP)                                GO TO 540
      KOUNT6 =   0
      KOUNT5 = KOUNT5 + 1
      IF (KOUNT5 .LT. N)                               GO TO 540
      IF (KI .LT. 2)                                   JJ = JJ+1
      GO TO 490
  440 IF (JB .EQ. JBPP)                                GO TO 460
      KOUNT6 =   0
      IF (F)                                           GO TO 480
      GO TO 500
  460 IF (F)                                           GO TO 540
      KOUNT6 = KOUNT6 + 1
      IF (KOUNT6 .LT. 2*N)                             GO TO 500
      F      = .TRUE.
      KOUNT2 =   0
      KOUNT5 =   0
      KI     =   0
      GO TO 540
  480 KOUNT5 = KOUNT5 + 1
      KI     = KI + 1
      IF (KOUNT5 .LT. N)                               GO TO 540
  490 F      = .FALSE.
      JBPP   = JB
      XYZ6   = BEXTPP
  500 IF (JB)                                          510,  650,  520
  510 XYZ6P  = AMAX1(XYZ6,BEXTP)
      XYZ6   = AMIN1(BEXTR,BEXTP,BEXTPP,BEXTP3,XYZ6)
      IF (JBPP)                                        650,  650,  530
  520 XYZ6P  = AMIN1(XYZ6,BEXTP)
      XYZ6   = AMAX1(BEXTR,BEXTP,BEXTPP,BEXTP3,XYZ6)
      IF (JBPP)                                        530,  650,  650
  530 CEXTP3 = CEXTPP
      CEXTPP = CEXTP
      CEXTP  = CEXTR
      CEXTR  = XYZ6P
      KOUNT5 = KOUNT5 + 1
      XYZ1   = CEXTR
      XYZ2   = CEXTP
      XYZ3   = CEXTPP
      XYZ4   = CEXTP3
      TCPPP  = TCPP
      TCPP   = TCP
      TCP    = TC
      TC     = TPREV
      TPPP   = TCPPP
      TPP    = TCPP
```

```
      TP      = TCP
      TRAV    = TC
      GO TO 560
  540 XYZ1    = BEXTR
      XYZ2    = BEXTP
      XYZ3    = BEXTPP
      XYZ4    = BEXTP3
      TBPPP   = TBPP
      TBPP    = TBP
      TBP     = TB
      TB      = TPREV
      TPPP    = TBPPP
      TPP     = TBPP
      TP      = TBP
      TRAV    = TB
      GO TO 560
C
  550 XYZ1    = EXTR
      XYZ2    = EXTRP
      XYZ3    = EXTRPP
      XYZ4    = EXTRP3
      TTPPP   = TTPP
      TTPP    = TTP
      TTP     = TT
      TT      = TPREV
      TPPP    = TTPPP
      TPP     = TTPP
      TP      = TTP
      TRAV    = TT
  560 D       = .FALSE.
  650 IF (C)                                          GO TO 780
      IF (B)                                          GO TO 660
      RX      = RAV
      KOUNT1 =   0
      B       = .TRUE.
      GO TO 690
  660 IF (ABS(RAV) .GT. 1.0)                          GO TO 665
      XY      = DELTAT*.01
      GO TO 670
  665 XY      = .001*ABS(RX)
  670 IF (ABS(RAV - RX) .GT. XY)                      GO TO 680
      KOUNT1 = KOUNT1 + 1
      IF (KOUNT1 - N)                                  690,  760,  760
  680 B       = .FALSE.
  690 IF (T .LT. TCRIT)                               GO TO 1150
      WRITE (6,3000) T,J,RP,RAV,RPP,RX,EXTR,EXTRP
      K1      =   2
      IF (F)                                          GO TO 710
      JZ      = JB
      JZP     = JBP
      GO TO 720
  710 IF (E)                                          GO TO 730
      JZ      = JS
      JZP     = JSP
  720 WRITE (6,6000) XYZ3,XYZ2,XYZ1
      IF (ABS(JZ) - ABS(JZP))                          740,  750,  750
  730 IF (ABS(RP - RAV) .GE. ABS(RPP - RP))           GO TO 750
  740 K2      =   3
      GO TO 1140
  750 K2      =   6
```

```
      GO TO 1140
  760 C       = .TRUE.
      TBIAS   = T
  780 IF (ABS(RAV) .LE. .01*ABS(XYZ1))                    RAV = 0.0
      BIAS    = RAV
      IF (T .GE. TCRIT)                                   GO TO 830
      IF (D)                                              GO TO 1150
      XEXTR   = ABS(XYZ1 - BIAS)
      IF (XEXTR .GT. 1.00001*EPSLON)                      GO TO 810
      KOUNT2 = KOUNT2 + 1
      IF (KOUNT2 - N)                                     1150,  880,  880
  810 KOUNT2 =   0
      GO TO 1150
  830 IF (KOUNT2 .LE. 0)                                  GO TO 850
      K2      =   3
      GO TO 875
  850 IF (XEXTR .LE. 0.0)                                 GO TO 870
      K1      =   2
      GO TO 890
  870 K2      =   6
  875 WRITE (6,4000) J,KOUNT2,T,BIAS,TBIAS,TRAV,TP
      K1      =   1
      GO TO 1140
  880 K1      =   1
  890 SLOPE1 = XYZ1 - XYZ3
      IF (ABS(SLOPE1) .LE. 1.0E-5*ABS(XYZ1))              GO TO 900
      IF (SLOPE1)                                          910,  900,  920
  900 M1      =   0
      GO TO 930
  910 M1      =   1
      GO TO 930
  920 M1      =   2
  930 SLOPE2 = XYZ2 - XYZ4
      IF (ABS(SLOPE2) .LE. 1.0E-5*ABS(XYZ2))              GO TO 940
      IF (SLOPE2)                                          950,  940,  960
  940 M2      =   0
      GO TO 970
  950 M2      =   3
      GO TO 970
  960 M2      =   6
  970 MM      = M1 + M2 + 1
      MN      =  SIGN(1.0,XYZ1-XYZ2)
      GO TO (980,990,1000,1000,1010,1000,990,990,1010),MM
  980 SLOPE   =   0.0
      GO TO 1020
  990 SLOPE   = -MN
      GO TO 1030
 1000 SLOPE   = +MN
      GO TO 1030
 1010 XKK     = K1*(1 - K1) + 1
      SLOPE   = SIGN(1.0,XKK)*9.9
      GO TO 1060
 1020 K2      =   9
      K3      =  10
      K4      =  11
      GO TO 1110
 1030 IF(K1 .GT. 1)                                       GO TO 1100
      IF (SLOPE .GT. 0.0)                                 GO TO 1060
 1050 K2      =   6
      K3      =   7
```

```
      K4      =    8
      GO TO 1110
 1060 K2      =    3
      K3      =    4
      K4      =    5
      IF (K1 .GT. 1)                                        GO TO 1110
      IF (T - TRAV .LT. TINCR)                              GO TO 1110
      WRITE (6,5000) T,TINCR
 1090 K2      =    6
      GO TO 1140
 1100 IF (SLOPE)                                            1060, 1060, 1050
 1110 IF (IPRINT .GT. 0)                                    GO TO 1130
      IF (K2 .LE. 8)                                        GO TO 1140
 1130 WRITE (6,2000) XA(K1),SLOPE,BIAS,T,XA(K2),XA(K3),XA(K4),XA(K1),
     1XEXTR
 1140 K1K2    = K1 + K2
 1150 RETURN
 2000 FORMAT (////20X10HSYSTEM IS A2,20HSTABLE  AND SLOPE = F6.1, 10H.
     1BIAS = G15.8,5X7HTIME = G15.8//45X10HSYSTEM IS 3A6,A2,6HSTABLE//
     250X8HXEXTR = G15.8)
 3000 FORMAT(////20X31HBIAS COULD NOT BE FOUND WITHIN G15.8,5H SEC.,5X,
     1I5,21H  EXTREMA WERE FOUND./15X5HRP = G15.8,5X6HRAV = G15.8,5X
     26HRPP = G15.8,5X5HRX = G15.8/40X7HEXTR = G15.8,5X8HEXTRP = G15.8//
     325X40HTHE SYSTEM IS CONSIDERED TO BE UNSTABLE.)
 4000 FORMAT (20X,I5,13H EXTREMA AND I5,46H STABILITY CURVE POINTS COULD
     1 BE FOUND WITHIN G15.8,4HSEC./15X,7HBIAS = G15.8,10H, TBIAS =
     2G15.8,10H, TEXTR = G15.8,19H, PREVIOUS TEXTR = G15.8//8X116HIF THE
     3 PERIOD IS RELATIVELY LONG, INCREASE TCRIT. ALSO, THE BIAS CONVERG
     4ENCE REQUIREMENT MAY BE MADE MORE STRINGENT.)
 5000 FORMAT (////31X7HTIME = G15.8,38H SEC.  NO EXTREMA HAVE BEEN FOUND
     1 FOR G15.8,5H SEC.//31X69HASSUME THE SYSTEM IS OVERDAMPED AND THER
     2EFORE UNCONDITIONALLY STABLE.////)
 6000 FORMAT (20X93HTHE SYSTEM IS BEATING OR HIGH FREQUENCY OSCILLATIONS
     1 OCCURRED. THE EXTREMA OF THE EXTREMA ARE/20X8HEXTPP = G15.8,10X
     27HEXTP = G15.8,10X7HEXTR = G15.8)
 7000 FORMAT (1X2A2,9A6)
      END
$DATA
   UN CONDITIONALLY     UNCONDITIONALLY   NEUTRALLY
```

APPENDIX C

RUNNING AVERAGE CONVERGENCE

Theorem: If $f(t)$ is a bounded real function defined on the interval $[0, \infty)$ and piecewise continuous, then the running average

$$RAV \equiv \int_0^T \frac{f(t)}{T}\, dt$$

must converge in the sense that

$$\lim_{T\to\infty} \frac{d}{dt}(RAV) = 0$$

Proof: The running average is differentiable; hence,

$$\frac{d}{dT}\int_0^T \frac{f(t)}{T}\, dt = \frac{f(T)}{T} - \frac{1}{T}\int_0^T \frac{f(t)}{T}\, dt \qquad \text{(B1)}$$

Consider the first term on the right side of equation (B1). To establish $\lim_{T\to\infty}$ of this function, let $C \geq 0$ be given. Then, since $f(t)$ is bounded over the interval, there exists a number $M > 0$ such that

$$|f(T)| \leq M \qquad \text{for } 0 \leq T < \infty$$

Now, for $T > 0$,

$$\frac{|f(T)|}{T} = \left|\frac{f(T)}{T}\right| \leq \frac{M}{T}$$

For any $C \geq 0$ there exists a number N such that, for $T > N \geq 0$,

$$\frac{M}{T} \leq C$$

or

$$\left|\frac{f(T)}{T}\right| \leq C$$

Thus,

$$\lim_{T\to\infty} \frac{f(T)}{T} = 0 \tag{B2}$$

Considering the second term on the right side of equation (B1), let a number $C > 0$ be given. Again since $f(t)$ is bounded over the interval, there exists a number M such that

$$-M \leq f(t) \leq M$$

It follows therefore that

$$-M = \int_0^T -\frac{M}{T}\,dt \leq \int_0^T \frac{f(t)}{T}\,dt \leq \int_0^T \frac{M}{T}\,dt = M$$

Thus,

$$\int_0^T \frac{f(t)}{T}\,dt$$

is bounded, and

$$\left|\int_0^T \frac{f(t)}{T}\right| \leq M \qquad \text{for } 0 \leq T < \infty$$

Now, for $T > 0$,

$$\frac{\left|\int_0^T \frac{f(t)}{T}\,dt\right|}{T} = \left|\frac{1}{T}\int_0^T \frac{f(t)}{T}\,dt\right| \leq \frac{M}{T}$$

Again for any $C \geq 0$ there exists a number N such that for all $T > N > 0$

$$\frac{M}{T} \leq C$$

or

$$\left|\frac{1}{T}\int_0^T \frac{f(t)}{T}\,dt\right| \leq C$$

Thus

$$\lim_{T\to\infty} \frac{1}{T}\int_0^T \frac{f(t)}{T}\,dt = 0 \qquad \text{(B3)}$$

Now, from equations (B1), (B2), and (B3) it follows that

$$\lim_{T\to\infty} \frac{d}{dt}\int_0^T \frac{f(t)}{T}\,dt = 0$$

which establishes the theorem.

Corollary: It logically follows then that if the running average

$$RAV \equiv \int_0^T f(t)dt$$

does not converge in the sense that

$$\lim_{T\to\infty} \frac{d}{dT} (RAV) = 0$$

and, if f(t) is a piecewise continuous real function defined on the interval $[0, \infty)$, then f(t) cannot be bounded over the interval.

This, then, establishes the concept that, if given a sufficient amount of time (infinite, if necessary) and if the running average does not converge to give the bias, then f(t) is unbounded and, therefore, unstable. In DIGSTA this is approximated by requiring convergence within TCRIT. (Since TCRIT can be picked to look like infinity for most practical systems.) The program further requires that

$$\frac{d}{dT} RAV = 0$$

for N consecutive times.

With slightly more difficulty, the following theorem can also be shown.

Theorem: If f(t), f(t), . . ., $f^n(t)$ are piecewise continuous, bounded real functions defined on the interval $[0, \infty)$, then the running average

$$RAV \equiv \int_0^T \frac{f(t)}{T} dt$$

must converge in the sense that

$$\lim_{T\to\infty} \frac{d^n}{dT^n} (RAV) = 0 \qquad \text{for } n = 1, 2, \ldots$$

This is, of course, a stronger convergence requirement. However, if this kind of convergence is not achieved, then either f(t) or one of its derivatives is not bounded (therefore unstable). This is a stronger stability statement. The convergence requirement of the program more closely resembles this requirement (because of the n consecutive dRAV/dt = 0 checks).

REFERENCES

1. Kaplan, Wilfred: Stability Theory. Symposium on Nonlinear Circuit Analysis. Vol. 2. Microwave Research Institute, Brooklyn, Arp. 1956, pp. 1-20.

2. Magiros, Demetrios G.: On Stability Definitions of Dynamical Systems. Proc. Nat. Acad. Sci., vol. 53, no. 6, June 1965, pp. 1288-1294.

3. Bellman, Richard E.: Stability Theory of Differential Equations. McGraw-Hill Book Co., Inc., 1953.

4. LaSalle, Joseph; and Lefschetz, Solomon: Stability by Liapunov's Direct Method, with Applications. Academic Press, 1961.

5. McCracken, Daniel D.; and Dorn, William S.: Numerical Methods and FORTRAN Programming, with Applications in Engineering and Science. John Wiley and Sons, Inc., 1964.

6. Wenzel, Leon M.; and Szuch, John R.: Analysis of Chugging in Liquid-Bipropellant Rocket Engines Using Propellants with Different Vaporization Rates. NASA TN D-3080, 1965.

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546