Abstract

The computer program LCAP2 (Linear Controls Analysis Program) provides the analyst with the capability to numerically perform classical linear control analysis techniques such as transfer function manipulation, transfer function evaluation, frequency response, root locus, time response, and sampled-data transforms. It is able to deal with continuous and sampled-data systems, including multiloop multirate digital systems, using s, z, and \( w \) transforms. This program has been used extensively for years at the Aerospace Corporation in the analysis and design of satellite and launch vehicle control systems.

I. Purpose

The program LCAP2 was designed to provide the control system analyst with most of the classical analysis tools needed for analyzing complex continuous and sampled-data systems by transform techniques. Primary considerations in the development of this program were ease of use and computational accuracy. A set of transfer function and polynomial operators has been defined in a fashion similar to the instruction set of a computer. Transfer function and polynomial arrays are defined to be referenced with indices so that they may be easily addressed by the operators. The combination of this set of LCAP2 operators and the form of the data structure provides a very flexible and easy to use program.

The LCAP2 program is the successor to LCAP [1] which is a batch program utilizing card inputs. This original version did not have the flexibility to allow the user to easily develop code to automate, for example, a complete stability analysis task beginning with the input of raw data to the generation of the stability plots. This is a very desirable feature in an industrial environment. The batch version of LCAP2 provides this flexibility since each LCAP2 operator is coded as a FORTRAN subroutine. An interactive version of LCAP2 is also available.

There are over seventy LCAP2 operators. A partial list of these operators is given in Table 1.

II. Language

The program is written entirely in FORTRAN with the exception of one subroutine which is written in assembly language.

III. Description and Capability

The data structure of the program includes (1) \( s \), \( z \), and \( w \) plane transfer functions designated as \( \text{SPTF}_i \), \( i = 1, 2, \ldots \), \( \text{ZPTF}_i \), \( i = 1, 2, \ldots \), and \( \text{WPTF}_i \), \( i = 1, 2, \ldots \), respectively, and (2) polynomials designated as \( \text{POLY}_i \), \( i = 1, 2, \ldots \). Operations on these transfer functions or polynomials are specified by references to their indices. For example, to add \( \text{SPTF}_1 \) to \( \text{SPTF}_2 \) and store the results in \( \text{SPTF}_3 \), the LCAP2 operation would be \( \text{SPADD}(3, 1, 2) \). In the batch version this operation is specified by the FORTRAN statement \( \text{CALL SPADD}(3, 1, 2) \). In the interactive version this operation can be \( \text{SPADD}(3, 1, 2) \) or, if the arguments were left out, the program will prompt the user for them.

The transfer functions are represented as ratios of polynomials. The user can load data into a transfer function using either the coefficient or the root form representation. An example is the operator \( \text{SPLDC}(i) \) which is an \( s \) plane load of coefficient data into \( \text{SPTF}_i \). The program can save both the coefficient and the root form representation for each transfer function so that arithmetic operations can be implemented with greater accuracy. For example, to multiply two transfer functions the product is obtained by collecting the roots of the numerator and the denominator beyond those common roots of the denominator, if there are any, are factored out before the addition and rationalization is carried out.

A typical use of these operators for a simple system would be to reduce a block diagram to a single open or closed loop transfer function using the add, subtract, multiply, and divide operators. Then one of the operators used to implement the classical controls analysis techniques can be applied. For example, if \( \text{SPTF}_1 \) is the open loop transfer, then the operator \( \text{SFREQ}(i) \) can be used to compute the frequency response so that the system can be evaluated.

For complex continuous systems, the above block diagram reduction method may not be practical to apply. Cramer's method for transfer function evaluation could then be used. For a set of Laplace transforms of differential equations represented in matrix form by

\[
\text{M}(s) = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}
\]

and \( \text{b}(s) = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \)

where \( \text{M}(s) \) is a matrix of polynomial coefficients, \( \text{X}(s) = \begin{bmatrix} x_1(s) \\ \vdots \\ x_n(s) \end{bmatrix} \), and \( \text{b}(s) \) is the vector relating the input \( u(s) \) to the set of equations. Cramer's method for computing the transfer function between \( u(s) \) and the \( j \)th element of \( \text{X}(s) \) is given by

\[
\frac{x_j(s)}{u(s)} = \frac{\det(M_j(s))}{\det(M(s))}
\]

where \( M_j(s) \) is equal to \( M(s) \) with column \( j \) replaced by the vector \( b(s) \). To apply this method the user first enters the data for the matrix \( M(s) \). Then the operator \( \text{DETRM}(i) \) is used. This will compute the determinant and store the resultant polynomial into \( \text{POLY}_j \). Next, the user enters the data for \( M_j(s) \) and then the operator \( \text{DETRM} \) is used with a different argument. If \( j \) is the argument, the resultant determinant is stored into \( \text{POLY}_j \). After both the numerator and the denominator determinants have been computed, they can be copied into \( s \) plane transfer function \( \text{SPTF}_i \) with the \( \text{CPYPS}(i,j,k) \) operator.

For sampled-data systems computation of the \( z \) and \( w \) plane transfer functions from the \( s \) plane description are provided by the operators \( \text{SZXFM}(i,j) \) and \( \text{SWXFMT}(i,j) \), respectively. Both of these transform operators allows the inclusion of time delay and the zero-order
hold. For transformation between the z and w plane, the bilinear transformation operators ZWFXM(i,j) and WZFXM(i,j) are available. Analysis of small order systems can be performed in either the z or w plane. The analyst may prefer to perform the analysis in the w plane since this would allow the use of the Bode design techniques. However, if the order of the system is high, the analysis must be performed in the w plane since z plane transfer function coefficients cannot be as accurately represented by the computer.

Several operators are available for use in analyzing multirate sampled-data systems. For the analysis of the fast to slow rate sampler where the ratio of the sampling rates is an integer n, Sklansky's [2] frequency decomposition method is implemented as the operators ZMFRQ(i,n) and WMFRQ(i,n). This decomposition method for the sampler in Fig. 1 expresses the transform of the slower output transform $C^T(z)$ in terms of the faster input transform $C^T(n(z^n))$ as:

$$C^T(z) = \frac{1}{n} \sum_{k=0}^{n-1} C^T(n(z^n)e^{\frac{2\pi k}{n}})$$

(3.3)

The user must first compute $C^T(z)$ and save it into ZPTF_i. Then when ZMRFQ(i,n) is called, the frequency response of $C^T(z)$ is computed. An example using this operator will be given later.

The program includes many features to reduce the amount of data which the user must enter. For frequency responses, automatic frequency selection is available which will choose all the frequency points required to produce a continuous smooth plot without skipping over lightly damped modes. Gain and 180 degree crossover frequencies are also automatically found so that the printout of the response will include the gain and phase margins. For root locus plots, gains are automatically incremented linearly or geometrically with provision for the user to specify some of the values. Automatic scaling for all plot variables is available.

The interactive version of the program provides extensive prompting so that a user's manual is not required. For the experienced user though, the amount of prompting may be excessive. Provisions are made to reduce the amount of prompting by allowing the user to enter some of the parameters directly without the use of prompts.

### IV. Limitations

Data structures for the transfer functions and polynomials require that the order of the polynomials be less than fifty. An arbitrarily large number of transfer functions and polynomials are available to the user since disk storage is utilized when the number becomes too large. The dimension of the matrix data used for transfer function evaluation must not be greater than 30×30 and the polynomial elements must be fourth order or less. Since the determinant of this matrix will be stored in one of the polynomials, the order of the resultant determinant polynomial must be less than fifty.

Although the data structure used for transfer functions limits the use of rational transfer functions to less than fifteenth order, the program has a limited capability to accommodate both higher order transfer functions and nonrational functions through the use of user supplied FORTRAN functions.

### V. Availability

When the user's manual of the batch version is completed (estimate is late 1982) both the manual and the source code will be available to agencies supporting DOD projects and studies. Requests are to be addressed to: Administrator, Information Processing Division, The Aerospace Corporation, P. O. Box 92957, Los Angeles, California 90009.

The FORTRAN programs have been compiled on the CDC (Control Data Corp.) FORTRAN EXTENDED 4 Compiler. The batch version runs on the CDC 7600 or 760 using the SCOPE 2.1.5 Operating System. The interactive version runs on the CDC 720 using the CDC INTERCOM Version 5.0 System. Batch jobs typically require 140K-240K words. No overlays are currently used. For the interactive version though, the segment loader is used since only 120K words are available to the user on the CDC 720.

The program will require some modification if it is to be operated on another system. Primary effort will be to replace the non-ANSI FORTRAN 4 ENCODE and DECODE statements. A future FORTRAN 5 version of this program will replace these statements with ANSI standard internal write and read statements. A FORTRAN version of the single subroutine written in assembly language is available, although it executes much more slowly.

### VI. Examples

**Example 1:** The FORTRAN code to compute the open loop frequency response of the sampled data system in Fig. 2 is to be determined.

The block diagram is labeled with LCAP2 s and z plane transfer functions to be used in the analysis. The FORTRAN code to compute the open loop transfer function, along with the description in quotes, is shown in Table 1.

The range of the frequency values to be used in computing the z plane frequency response is specified in the s plane. The program will perform the conversion to the z plane when evaluating the response. Four
values of $\Omega$ were used for this example to specify the frequency points. Since the program will dynamically choose its own frequency points for computing the response to ensure that a continuous smooth plot is produced, it most likely will not select the frequencies of 0.1 and 1.0. The inclusion of frequency points between $\Omega(1)$ and $\Omega(N_{\Omega})$ by the user allows the explicit specification of some of the values to be used in computing the response.

Since user input for the batch version of LCAP2 is FORTRAN code, the user can easily perform tasks such as parametric or sensitivity studies with the use of FORTRAN DO loops, or implementation of algorithms to select design values.

**Example 2:** The use of the LCAP2 operators ZMRFQ(i,n) and ZVCNG(i,j,n) will be applied to the analysis of the system in Fig. 3.

The open loop frequency response of the outer loop with the inner loop closed will be determined. The first part of the analysis is to simplify the block diagram. For the forward loop at the slower sampling rate, the transform

$$D_1 T(z)^* H T/n(z_n)$$

(6.1)

can be simplified if $D_1 T(z)$ is first expressed in terms of the faster $z_n$ variable so that transfer function multiplication can be applied. The operator ZVCNG(i,j,n) provides this capability. If $D_1 T(z)$ was loaded in ZPTF1, the operator ZVCNG(2,1,n) will transform $D_1 T(z)$ to the faster rate by replacing $z$ with $z_n$ and storing it into ZPTF2.

If $H T/n(z_n)$ is loaded into ZPTF3, the operator ZPMPY(4,2,3) will compute the desired transform and store the results into ZPTF4. The faster inner loop can be simplified to a single transfer function by first computing the $z$ transform of $G_2(s)$ with the zero order hold and then computing the closed loop transfer function using block diagram reduction. If this closed loop transfer function is stored in ZPTF5, the block diagram can be simplified as shown in Fig. 4.

The inclusion of the fictitious $T/n$ sampler after $G_1(s)$ does not affect the sampling process. It does, however, simplify the analysis. If the $z$ transform of $G_1(s)$ with the zero order hold is computed and stored, the product of this transfer function with ZPTF4 and ZPTF5 can be computed. If this result is stored in ZPTF6, the desired open loop transfer function is given by Fig. 5.
By the frequency decomposition method the output at the slower rate, \( C^T(z) \), is given by

\[
C^T(z) = -\frac{1}{n} \sum_{k=0}^{n-1} C^T(z_n^k z_n^{2\pi k/n})
\]

(6.2)

The output at the faster rate, \( C^{T/n}(z_n) \), is given by

\[
C^{T/n}(z_n) = G_{10} T^{n}(z_n) \ast E^T(z)
\]

(6.3)

If \( E^T(z) \) is expressed in terms of the faster \( z_n \) variable as

\[
E^{T/n}(z_n^n) = E^T(z)
\]

(6.4)

substitution of Eq. (6.3) in Eq. (6.2) yields

\[
C^T(z) = \frac{1}{n} \sum_{k=0}^{n-1} G_{10} T^{n}(z_n^k z_n^{2\pi k/n}) \ast E^T(z)
\]

(6.5)

The open loop transfer function is thus given by

\[
C^T(z) = \frac{1}{n} \sum_{k=0}^{n-1} G_{10} T^{n}(z_n^k z_n^{2\pi k/n})
\]

(6.6)

Since \( G_{10} T^{n}(z_n) \) is stored in \( ZPTF_{10} \), the operator \( ZMRFQ(10,n) \) can be used to evaluate the open loop frequency response.

VII. References


Eugene A. Lee was born in Sacramento, California on July 26, 1939. He received his B.S.E.E. and M.S.E.E. degrees from the University of California at Berkeley in 1962 and 1963, respectively. From 1963 to 1966 he was a research engineer with North American Aviation, Space and Information Division. Since 1966 he has been with the Aerospace Corporation where he is currently an engineering specialist. His area of interest includes linear systems, digital control and computer-aided analysis of control systems. He is a member of the Tau Beta Pi, Eta Kappa Nu and IEEE.