NASA TECHNICAL MEMORANDUM

NASA TM X- 67908

NASA TM X-67908

N72-36189 (ACCESSION NUMBER)	(THRU)
31 (PAGES)	(CODE)
TM-X-67908 (NASA CR OR TMX OR AD NUMBER)	G3 08 (CATEGORY)

FACILITY FORM 602

APPLICATIONS OF SYMBOLIC COMPUTING METHODS TO THE DYNAMIC ANALYSIS OF LARGE SYSTEMS

by Carl F. Lorenzo and John P. Riehl
Lewis Research Center
Cleveland, Ohio

TECHNICAL PAPER proposed for presentation at
Fifth Congress of the International Federation
of Automatic Control
Paris, France, June 12-17, 1972

31

ABSTRACT

APPLICATIONS OF SYMBOLIC COMPUTING METHODS TO THE DYNAMIC ANALYSIS OF LARGE SYSTEMS

By Carl F. Lorenzo and John P. Riehl

Lewis Research Center
Cleveland, Ohio 44135

E-6519

The dynamic analysis and control of very large systems yet remains a problem for the analyst. The tools currently available to him rely largely on matrix methods. Analysis of such problems in the missile field, specifically the Pogo problem, which includes the structural, fluid, engine, and control dynamics of large multi-stage booster vehicles, was the source of a somewhat different approach to the analysis of large systems. In general, it is recognized that most man-made systems, either physical or otherwise, tend to be not highly interactive. That is, most of the elements are connected only to other elements near to it rather than inter-connected with all other elements which compose the system. The use of matrix techniques to analyze such problems is found lacking because a large number of matrix elements are zero.

A new approach to the problem lies in the use of symbolic computer methods applied directly to the system equations. This study will discuss some approaches to application of symbolic computing methods to the analysis of very large physical dynamic systems. The nature of the symbolic approach together with the achieved and potential benefits will be discussed.

This paper further studies two specific techniques applied to the analysis of large dynamic systems. Since the symbolic computing language is very well suited to the operations with algebraic equations, both techniques use the transfer function concept as a tool for the analysis of large linear dynamic systems. Also, both techniques have been coded in the experimental symbolic computer language FORMAC. (Formula Manipulation Compiler). The first of these approaches, REDUCE I, establishes the techniques and a computer program to symbolically reduce arbitrary block diagrams associated with large systems for desired transfer functions. Symbolic closed form solutions are determined in several forms including an expanded form in terms of the driving frequencies and system constants. Along with this first technique, programs are also written to numerically evaluate the symbolic solutions. The program has been applied to several research problems which include both lumped and distributed parameter systems. The distributed parameter forms are built into the program, and are handled automatically.

A second computer program, REDUCE II, is also based on the use of symbolic computing methods and has been written to accommodate large engineering systems. REDUCE II symbolically calculates the transfer functions of any linear block diagram output variable to any or all input variables. The solution using this technique is presented in the compact form of a set of nested functions. The program can handle systems as large as 600 equations (of essentially any order) and is intended as a tool for the analysis of complex control and dynamic systems. A sister FORTRAN program evolved to numerically evaluate solutions formed by REDUCE II is used to obtain amplitude ratio and phase angles as function of frequency.

The advantage of the symbolic approach to large systems over the matrix approach is entailed in the fact that the storage required in the symbolic approach is proportional to the number of variables, hence, the number of equations, as opposed to being proportional to the square of the number of variables as in the matrix methods. Applications of both techniques discussed above are presented in the subject paper. The paper also projects other areas of application of the symbolic method.

APPLICATIONS OF SYMBOLIC COMPUTING METHODS TO THE DYNAMIC ANALYSIS OF LARGE SYSTEMS

By Carl F. Lorenzo and John P. Riehl

INTRODUCTION

The dynamic analysis and control of very large systems remains a problem for the analyst. Indeed, in addition to the increased complexity and size of most physical systems, there is an increasing awareness of large system problems in the "softer" sciences. Some of the current and projected applications of the systems approach in these areas include: (1) Management systems; (2) Various Physiological Systems; for example, circulatory, respiratory, etc.; (3) Urban Dynamics and Planning; (4) Economic Systems; (5) Industrial Dynamics; (6) Environment Pollution; (7) Population Growth (where the entire world is considered as a system considering such states as number of people, food, resources, and waste) and of course the various distribution systems; (8) Transportation, Communications and Power Networks. Hence, little justification is needed to study the large system problem.

Any serious approach to the large systems problem will center around the computer, with its ability to handle routine tasks quickly and to hold large quantities of information available in memory for ready access. Accepting this premise, the central problem then becomes determination of the best way in which to utilize the computing power available at any given time.

In this paper a somewhat unconventional approach to the analysis of the dynamics of large systems will be considered. The basis of this approach is the use of symbolic computing methods which are applied directly to the system equations.

The scope of the paper will include a brief discussion of the large system problem, some background on symbolic computing methods, and two programs applying the symbolic technique to dynamic analysis of systems will be considered. Further, some attempt will be made to project additional areas of application for the method.

THE LARGE SYSTEM DYNAMICS PROBLEM

Most large systems and physical large systems in particular, have a number of properties which are important in determining how they should be treated. Large systems are *complex*. They are complex because: (1) the *elements* that compose them *are complex*. (2) The *elements* that compose them *are many*, that is, the systems are by definition, large, and (3) the *arrangement of the elements* may be complex. In addition, the mathematical description may increase or decrease the apparent system complexity. Large systems are usually *nonlinear*. In most physical cases they are *distributed*. If our knowledge about them is incomplete they are *non-deterministic*. Finally, a very important property, large systems generally are *not highly interactive*; that is, the elements usually do not communicate with all the other elements of the system but rather only a few, generally those which are nearby.

The fundamental dynamics problem can be described as: Prediction of the response of any desired system state to any control or disturbance input or combination of inputs. This differs from the controls problem which is to determine what can be done to certain available inputs to achieve desirable behavior in all parts of the system. It differs also from large systems considerations which seek to determine the desired communication paths between the elements, for example, the layout of a telephone system. Fundamentally, the dynamics problem assumes a system arrangement and seeks to describe it in a concise manner.

General mathematical descriptions of most physical system dynamics are provided by systems of nonlinear partial differential equations. The analyst nearly always linearizes, usually applying perturbations methods. Also each partial differential equation is usually approximated by a system of ordinary differential equations (lumping). This has the effect of introducing intermediate states which increase systems size and complexity of arrangement.

At the heart of systems problems is the question of organization; that is, the functional interrelationship of the elements forming the system. The analysis part of the dynamics problem is to determine the relationship of any desired output state to any or all input states, regardless of arrangement and nature of the elements and independent of the remaining system states. In the frequency domain, this is the determination of the desired transfer functions. Thus, the transfer function can be considered as a basic large systems tool which determines the relationship of the desired state to a particular input. The systems problem mathematically, is a topological problem and for linear Laplace transformed systems the dynamics problem is an algebraic one.

How the system analysis is done, of course, depends on the mathematical tools available. In general, in dealing with systems of any size at all, the use of the computer is a must. The traditional tools which have been used are based on two fundamental ideas. The first of these is simulation. Simulation has the advantage of great generality in the systems which can be handled. Both linear and nonlinear systems can be handled and approximations to distributed systems can be made. One can be certain that a solution to a large systems problem can be achieved if enough equipment or computing time is available. The weakness of the simulation process, however, is that it is very difficult to generalize the results.

The second approach to systems analysis entails a wide variety of methods centered around the use of matrix techniques. The matrix approach supplies a fundamental system organizational tool. The approach has the advantage of a significant theoretical background which can be applied to solving the systems problem. However, the matrix method is not really designed for large systems. This can be argued by referring to figure 1. Figure 1(a) shows a schematic diagram of sixteen elements arranged in a four by four array. The elements are allowed to communicate with each other on a vertical and horizontal arrangement. Using matrix techniques to describe this communication, a matrix of the form of figure 1(b) is obtained. In this matrix, the ones indicate communication of the column elements with the row elements. Communication in both directions is assumed and self-communication is not considered. In this particular case, namely a four by four arrangement (M = 4) it can be seen that the matrix is rather sparse. Indeed, since no element communicates with more than four other elements it is apparent that as the system size increases the relative number of zeros in the matrix will increase. This is shown by the graph (fig. 1(c)) which plots the number of zeros in the matrix divided by the number of matrix elements as a function of system size. In the limit, as system size increases to very large systems, this ratio approaches one. In particular, less than one percent of the elements are non zero for M = 20. If diagonal communication is allowed, each element communicates with as many as eight other elements. The situation becomes somewhat worse in that M = 30 is required to achieve one percent non zero elements in the communications matrix. The agrument posed here is that mentioned earlier, namely, that most physical systems and most man-made systems in particular are not highly interactive. One can of course describe very large systems using matrices and then resort to sparse matrix methods to analyze them. However, serious questions arise as to the value of using the matrix format in the first place. Indeed, if one is to consider a system composed of a thousand elements, he must be ready to handle matrices, with a million elements, and to manipulate them and calculate with them. Admittedly, many short cut procedures have been arranged for the matrix methods. However, it may be that a different approach is required or at least desirable.

Another approach to large system analysis lies in the direct use of the algebraic equations (frequency domain). In working directly with the algebraic equations, we are working with a system of N elements or at least proportional to N elements rather than proportional to N^2 elements, as in the case of the matrix approach. This can be done by hand, and classically has been done in this manner; that is, the manual reduction of block diagrams. However, for very large systems, this becomes a very tedious and error prone task. With the emergence of symbolic computing languages, this process can be done in an automated manner using the digital computer, allowing elimination of a routine task and potential errors. In addition it allows the use of a new organizing tool, the symbolic language itself.

In the work that follows, direct solution of the system equations (frequency domain) for desired transfer functions will be referred to as block diagram reduction. The block diagram is important because it gives a visual picture of the relationship between the important variables; hence, the elements of the system. Before discussing the question of linear block diagram reduction it is appropriate to discuss at least briefly, the nature of the symbolic computing languages.

SYMBOLIC ALGEBRAIC MANIPULATION

Computer algebraic manipulation means: the use of a computer to operate on mathematical expressions in which not all the variables are replaced by numbers, and in which some meaningful mathematical operation is to be done, ref. 1. The purpose of a manipulative language is to eliminate tedious mechanical algebra and the errors that can result in such algebra. Most manipulative languages are written as supersets of some other compiler level language. This gives algebraic languages the benefits of their subsets.

FORMAC may be taken as a typical example of an algebraic manipulation language. Originally, it was written as a superset of FORTRAN, and later as a superset of PL/I. FORMAC permits operations of addition, subtraction, multiplication, division, and exponentiation to be applied symbolically. Limited trigonometric functions and analytic differentiation are also provided. The processes of substitution, removal of parentheses, automatic simplification, comparison of expressions and more can be performed (details, ref. 2).

FORMAC functions as a translator from FORMAC notation into FORTRAN notation and FORTRAN subroutine calls. It builds symbol tables of expressions and tables relating FORMAC to FORTRAN variables. Certain additional nonexecutable FORTRAN statements are also created.

The FORMAC symbol table consists of two words. The first is the BCD name. The second word may be a FORTRAN variable or encoded information to the supporting subroutines if the symbol is algebraic. This encoded information identifies a variable as either atomic or let and supplies a pointer to the location (atomic) or expression (let) associated with the symbol. FORMAC expressions are encoded in delimited forward POLISH notation. The expression table is packed with an end of expression marker, and every expression begins a new word. Constants appear literally.

The limitations of FORMAC include: lack of integration capability and trigonometric identities. Further, expressions are not always as simplified as the user might like. This can lead to expressions that are intractible for large problems. The FORMAC operating system tends to be somewhat slow in comparison to normal FORTRAN programs. It should also be noted that for very large programs, storage (in terms of free list available to the FORMAC system) becomes a premium. The techniques in this paper were written in FORTRAN - FORMAC for the IBM 7094 - 7044 D.C.S. running under IBSYS. In spite of these limitations the symbolic computing approach forms a powerful tool which can be applied to a wide variety of problems. In the following sections FORMAC will be applied to reduction of block diagrams.

TOTAL SYMBOLIC REDUCTION PROGRAM - REDUCE I

A program (called REDUCE I) has been written in FORMAC, to symbolically reduce linear controls and dynamics block diagrams, ref. 3. The program symbolically manipulates the equations representing the block diagram to solve for the transfer function of a desired output variable to a desired input variable. The REDUCE I program performs four major steps: (1) reduction of the system of equations (of the block diagram) to one equation containing the two variables of the desired transfer function. (2) Solution of this equation for the transfer function in terms of the G functions (component transfer functions), (3) Substitution of the component information for the G functions. (4) Expansion of this equation and collection of terms of the real and imaginary parts.

The fundamental structure of the program is indicated by the simplified flow chart presented in figure 2. The user input into the program is minimal requiring first an indication of the numbers of: variables, equations,and inputs. A check is made of this information to assure that the size limitations of the program are not exceeded. The user must further specify the transfer function desired, the inputs of the block diagram,and the algebraic equations representing the block diagram. The block diagram variables are specified as subscripted X's. Other than this, the form is any linear combination of the X's and coefficients, namely, the G's or transfer functions of the individual components. If expansion of the G's is desired the component information must also be supplied. The G's are expressed in terms of the Laplace variable S and the various time constants and natural frequencies of the system. These data are sufficient to allow reduction to a symbolic form.

Before the actual reduction can take place, the order in which the variables are to be eliminated and equations to be used for the eliminations are determined by an algorithm built into the program. The first variable to be eliminated is solved for in the equation indicated by the order determination. This variable is then substituted into <u>all</u> other equations of the set containing that variable, thereby eliminating it. This process is continued for all the remaining variables except the two variables involved in the desired transfer function. The final equation contains only the variables of the transfer function and the G functions representing the system components.

The reduction method used in the program can be depicted using the sample system as in table I. The result of the reduction for the example is,

$$\frac{X_1}{X_2} = \frac{G_1G_3 + G_2G_3}{1 + G_2G_3} \tag{1}$$

This is the so-called G-form of the transfer function solution. The G functions can be expressed as $G_i = N_i/D_i$. This ratio is substituted into equation (1) which results in

$$\frac{X_1}{X_2} = \frac{D_1N_2N_3 + D_2N_1N_3}{D_1D_2D_3 + D_1N_2N_3} \tag{2}$$

for the example taken. In this form, the component information is easily introduced into the result. Assuming that $G_1 = \frac{1}{\tau S + 1}$, $G_2 = 0$, $G_3 = \text{SINH (1)}$. These G functions are then substituted into equation (2), thus

$$\frac{X_1}{X_2} = \frac{\text{SINH (1)}}{\tau S + 1} \tag{3}$$

This is the S form solution of the transfer function (unexpanded). From this form numerical evaluation is possible or further symbolic reduction can be done. For the symbolic result, S is replaced by $i\omega$ and powers of i are simplified to give

$$\frac{X_1}{X_2} = \frac{\text{ReS I N H (1)} + i\ \text{ImS I N H (1)}}{1 + i\,\tau\omega} \tag{4}$$

This is called the complex rational form. That is, the form

$$\frac{X_a}{X_b} = \frac{A + iB}{C + iD} \tag{5}$$

where A, B, C, and D are functions of the excitation frequency and the system parameters τ's, ω_n's, K's, etc. Availability of this symbolic form of the transfer function is very important in many research studies. The transfer function can also be numerically evaluated from this form.

A numerical evaluation program also has been written which can be applied either to the S form or the complex rational form to generate frequency response curves; amplitude ratio, phase angle as function of frequency.

The REDUCE I program capability is demonstrated with the following distributed dynamic application. The physical system to be analyzed is shown schematically in figure 3A. This system is composed of a main feed line feeding three subsidiary ducts. The dynamic problem is to determine the response of pressure in the subsidiary ducts, to pressure disturbances at the far end of the feeder duct; for example, to determine the frequency response of P_c/P_o. The system block diagram is shown in figure 3(b). The contents of the blocks, because each line is a distributed system, are hyperbolic functions of square root arguments. The form of these G's represents "four" terminal network solutions to the wave equations for each section of line.

The system equations are

$$
\begin{array}{llll}
X_2-G_1X_1 = 0 & X_6-G_6X_4 = 0 & X_{14}-G_{11}X_{11} = 0 & X_6-X_5-X_{11} = 0 \\
X_3-G_2X_{20} = 0 & X_7-G_7X_{15} = 0 & X_{15}-G_{12}X_{12} = 0 & X_8-X_7-X_{12} = 0 \\
X_{19}-G_3X_4 = 0 & X_8-G_8X_4 = 0 & X_{16}-G_{13}X_{13} = 0 & X_{10}-X_9-X_{13} = 0 \\
X_{18}-G_4X_{17} = 0 & X_9-G_9X_{16} = 0 & X_2-X_3-X_4 = 0 & X_{14}+X_{15}+X_{16}-X_{17} = 0 \\
X_5-G_5X_{14} = 0 & X_{10}-G_{10}X_4 = 0 & X_{19}+X_{18}-X_{20} = 0 &
\end{array}
$$

where the block diagram input is X_1, and X_{12}/X_1 is the solution required. The component information is indicated on the block diagram (fig. 3(b)) where

$$\gamma_n = \sqrt{L_nC_nS^2 + R_nC_nS} \qquad Z_{on} = \frac{1}{S}\sqrt{\frac{L_n}{C_n}\left(S^2 + \frac{R_n}{L_n}S\right)} \quad \text{and} \quad Z_m = \frac{R_m}{R_mC_mS + 1}$$

The transfer function solution for P_c to P_o in the G form is presented as equation (A1) in Appendix A. The transfer function in the S form with the component information substituted in for the N's and D's is equation (A2). These equations are presented to give an appreciation for the nature of a symbolic solution of a fairly sizable problem. This particular problem was solved in about one minute of computing time. The printed solution shown here is in FORTRAN notation. A deck of punched cards is also output which contains the symbolic solution. Typical numerical use of such information by evaluation for a set of conditions, for example, those shown in table II, is indicated in figure 3(c) where the transfer function has been evaluated for various values of the terminating characteristics of line B.

The advantages of this technique are: (1) the results are given in three symbolic forms G, S, and complex rational, and (2) the input is extremely simple and is in normal algebraic language (arrays and matrix manipulations for each frequency point are not required); (3) the symbolic solution need be generated only once and then re-evaluated for different frequencies or different component parameters. Also both distributed or lumped systems can be handled with ease. The computing time is small, even for large systems, and the user preparation time is minimal.

The most serious limitation of the work is the computer storage required. Symbolic transfer functions about twice the size of this example are the limit of the IBM 7094 - 7044 D.C. system. The FORMAC routine itself requires a large amount of storage and as system size increases the transfer function equations grow very large, burdening storage capabilities. Further, one of the main values of a symbolic solution, that of being able to look at the terms and make judgments concerning the solution, diminishes as the system size grows. For these reasons an attempt was made to handle very large systems for engineering problems using a somewhat different conceptual basis. This is the REDUCE II program.

REDUCTION OF VERY LARGE SYSTEMS USING THE REDUCE II PROGRAM

The basic philosophy of the REDUCE II program (ref. 4) is to avoid any steps which promote significant expansion of the forms within the computer and to handle the equations in such a manner that the system collapses rather than grows in the computer. That is, as variables are eliminated in the reduction process, the coefficients of the remaining variables will become combinations of the original coefficients (G's). As this occurs, the combinations will be replaced by single element coefficients (called super G's) and the mathematical equations defined by the substitution will be immediately output from the computer.

Mathematically, through the reduction and substitution process, a set of linear equations in the variable X with coefficients (G's) (considered to be constant) is transformed into an equation defining the transfer function (in terms of super G's) and a set of equations defining the super G's (called the solution set). Even within this framework, a large number of ways exist to proceed since the form of the equations composing the solution set has not been stipulated.

Some constraints on the form of the solution set equations can be projected from the intended use however. Such considerations dictate the following desirable properties of the solution set: (1) the super G's formed should be nested, that is, each G should depend only on the G's which preceed it and the original G's of the block diagram. (2) The likelihood of a zero divide occurring when the solution set is evaluated should be minimal. (3) The solution set should not require scaling. (4) The solution set should be of minimal size. Consideration of these properties, in particular (2) and (3), strongly suggest that a most desirable form for the super G's composing the solution set, should be

$\frac{\Sigma \prod G_j}{\pm 1 \pm \Sigma \prod G_i}$ or as close to this form as possible. This form is characterized by the fact that if the G terms in the numerator and denominator are of the same order the super G will approach either zero or one as frequency increases. Also because of the unity term in the denominator the possibility of a zero divide condition (by allowing any

G = 0) is null. The G = 0 special case is important since it is a primary method of simplifying block diagram topology. It appears that super G's with the unity additive term in the denominator can always be made to occur if the block diagram is reduced by loops. In addition, the general form approaches that desired. That this kind of solution form is possible, even for loops embedded in large block diagrams (ref. 4), has been used as a fundamental premise in the structure of the REDUCE II program.

The logic and flow of the REDUCE II program is illustrated in the flow chart of figure 4. The program inputs are: the solution form desired; the variables which are the block diagram inputs; the transfer functions desired; the symbolic equations of the block diagram; and an order string indicating when each variable is to be eliminated and when a substitution is to be performed. Counts of numbers of variables, equations, and G's, are made and compatibility checks are performed to insure in as much as possible that the equations of the system are solvable.

Following these checks a variable to be eliminated is determined from the order string. An appropriate equation containing the variable is found by the program and the algebra involved in eliminating that variable is performed symbolically. After all the variables of the particular sub-string of the order string have been eliminated the super G substitution procedure is performed. In making the super G substitution, the coefficients of the X's of all the affected equations are searched for the form $\pm 1 \pm \Sigma \Pi G_i$. If the form occurs the equations are divided through by it forming super G's of the type indicated earlier (called Natural form). If the form does not occur (for example, if reduction by loops is not used) it can be artifically introduced by adding 1 to an appropriate coefficient to achieve super G's with denominators of the $1 \pm \Sigma\Pi G_i$ form. This is a separate program option (called Artificial form). Following this division by the appropriate coefficient, the new coefficients of the X's have super G's substituted for them and are output from the program as they occur. This is repeated until all substrings have been eliminated. When this is done, the reduction is complete and the transfer function(s) can be formed. In this program, the transfer function for any output to all the block diagram inputs are formed simultaneously. When all the individual transfer functions have been formed, a check is made to assure the resulting equations are empty. That is, that there are no residual terms left in the equations which would indicate an error in the formulation of the system equations.

The user does not specify which equations are to be used in the reduction, but merely which variable is to be eliminated in a given step. The simplest equation (fewest variables) containing the variable to be eliminated is used to eliminate it. If the variable due for elimination occurs in more than two equations an algorithm attempts to identify the equations associated with block diagram loops containing either: first the variables involved in the order substring or second, any other variable yet to be reduced. If neither of these is possible then the simplest equation is used.

A numerical evaluation program (EVAL II) geared to the REDUCE II output has also been written. The program takes the solution set generated by REDUCE II, supplies the component information in the form of the normal G's and uses complex arithmetic to evaluate the super G's and transfer functions. The output of EVAL II consists of a tabulation of the transfer functions for each frequency point. Plot routines are also included.

The program function is best demonstrated by a simple application. The block diagram of figure 5 represents a system for control of the position of a shock wave in a supersonic inlet for a jet engine using engine fuel flow. Using "modern" control techniques a controller was designed to minimize the occurrence of inlet unstarts in the presence of a stoichastic disturbance. The control designer wishes to obtain specific transfer functions from which system frequency domain behavior could be studied. Since a PADE' approximation to the inlet plant dead time had been made for control design purposes, it was important to determine the effect of this approximation on dynamic performance. Transfer functions X_7/X_1, X_7/X_2, X_7/X_3 were obtained for this reason.

The partial results of the reduction (using the Artificial form solution) are given in the solution set A3. From these results, evaluations with different plants and controllers can be made. Essentially, any simplification of the topology and any complication of the components, is possible. The time required to execute this reduction was 25 seconds.

A numerical evaluation for the G's (component information) of Table III was made using the EVAL II program. The frequency range of interest was from 0.01 to 4.0 Hertz (due to user scaling, frequency x 100) by steps of 0.01 Hertz. The execution time required to evaluate these 400 points plus an additional 600 points for the automated plot routine for all three transfer functions was 32 seconds. Plots of typical results for the X_7/X_1 transfer function are presented in figure 6. The difference in the plots is of course the effect of the PADE' approximation at the higher frequencies. Systems similar to this with 10th order plants and controllers have also been analyzed.

Program capability was studied by reducing two and four terminal ladder networks. Systems as large as 600 equations were reduced in this study. The reduction rate is sensitive to the problem size as shown in figure 7.

Aside from storage and time limitations, the only problem encountered with the approach has been occasional numerical evaluations difficulties (for wide frequency band) for large highly interactive systems. These can generally be eliminated by proper selection of the reduction order string.

CONCLUSIONS AND PROJECTIONS

Two programs have been written to symbolically reduce controls and dynamics block diagrams. The first of these programs, REDUCE I, generates a total symbolic solution, and is applicable to derivational and research problems, where such solutions are needed for further analysis. The solution forms are: the G form, the S form, and the complex rational form. The program is capable of handling arbitrary linear block diagrams, but is limited to systems involving 30 equations and as many as 63 variables. The computing time required is modest for systems of these sizes and the user preparation time is minimal. The system can be either lumped or distributed parameter.

A second program, REDUCE II, was written to handle very large block diagrams as would be encountered in engineering applications. With this program transfer functions can be determined for any output variable to any or all input variables. The solution is determined as a set of non-linear algebraic equations. These equations, called super G's, are defined in terms of the original system G's and the super G's which preceed it. The solution set relations are output from the computer as they are formed thereby minimizing the critical problem of computer storage. The nested form of the solution set allows direct numerical evaluation. The premise of the program, is reduction of the block diagram by loops, which alleviates scaling and zero divide problems. The advantage of this program over previous techniques, is that the arrangement of the input data in the form of arrays and the matrix manipulations for each frequency are not required. The symbolic solution set need be generated only once and then evaluated for desired frequencies and G's.

The REDUCE II program allows solution of systems involving up to 600 equations with a total of 1200 G's (normal plus super G's). The input information required is nominal and the manner in which the reduction is performed is controlled by the user in stipulating the order string. The computing time required for the program varies with the size of the problem being solved. Typical rates are 0.6 of a variable per second for small problems to an initial rate of 0.1 variable per second for larger problems.

The symbolic computing technique appears to have significant potential for dealing with large systems dynamic problems. The symbolic approach supplies an organizing function. This can be used to advantage by allowing storage, proportional to the number of variables to be handled rather than the number of variables squared as is usually the case in matrix methods. Further, the symbolic method allows human knowledge and experience to be readily applied to the large system problem (i.e., reduction order string for REDUCE II).

One of the most important aspects of the symbolic computing method is the fact that once a topology is solved it is not necessary to solve it again, and it can then be evaluated for any specialized set of component information to generate desired sub-applications.

The weaknesses of the symbolic method lie in the pattern recognition area. That is, the ability to factor and the ability to integrate. Further, with current computers, the storage required for the program itself and the memory required for computed solutions are a limitation. However, as computers grow, this limitation should be reduced.

Extensions of the work presented in this paper would take the form of pre and post processing programs. Two preprocessing programs in particular would be of interest. It should be possible to write a symbolic program to automatically linearize a wide variety of nonlinearities. Such a program would accept the symbolic equations defining a system together with a nominal operating point and transform these into a linear system.

A second program could be written based on graph theoretical concepts to determine best strategies for the block diagram reduction for complex diagrams. This would likely be a non-matrix approach using tree structures. The result would be in terms of an order string to guide the reduction process. Further research would be required into the question of best solution form.

The post processing possibilities are even more interesting since the symbolic solution forms are available. Programs can be written to determine stability as a function of several system parameters. Using the transfer functions as generated by REDUCE II, determination of the zeros of the denominator and the infinities of the numerator would be required.

Symbolic programs can be written to examine the influence of key parameters on system dynamics. This would be done by numerically specializing all parameters but those of interest; the resulting expression would be one of an influence function character.

Programs to determine transient response might also be feasible. This would require an augmented block diagram to be generated and reduced and numerical inversion of the frequency response.

In the area of optimum controls the organizational aspects of the symbolic method should be useful. The differentiation capability of the language could be used to generate the Euler-Lagrange Equations in an automated manner. Integration of the resulting boundary value problem for a general case would be very difficult. However, special cases where the system characteristics are restricted might be handled by specially written symbolic integration algorithms.

REFERENCES

1. Sammet, Jean E.: Programming Languages: History and Fundamentals. Prentice-Hall, Inc., 1969.

2. Bond, Elaine R.: User's Preliminary Reference Manual for FORMAC. Boston Advanced Programming Dept., IBM, Feb. 1964.

3. Lorenzo, Carl F.; and Swigert, Paul: Computer Program for Symbolic Reduction of Block Diagrams Using FORMAC. NASA TN D-4617, 1968.

4. Lorenzo, Carl F.; and Riehl, John P.: Computer Program For the Reduction of Very Large Block Diagrams (Using FORMAC). Proposed NASA Technical Note.

APPENDIX: Solutions of Applications

X_{12}/X_1=

```
THE NUMERATOR
     G(1)*G(5)*G(8)*G(9)*G(11)*G(13)+G(1)*G(5)*G(8)*G(11)+G(1)*G(8)+G(1
     )*G(8)*G(9)*G(13)

THE DENOMINATOR
     G(2)*G(3)+G(2)*G(3)*G(5)*G(7)*G(9)*G(11)*G(12)*G(13)+G(2)*G(3)*G(5
     )*G(7)*G(11)*G(12)+G(2)*G(3)*G(5)*G(9)*G(11)*G(13)+G(2)*G(3)*G(5)*
     G(11)+G(2)*G(3)*G(7)*G(9)*G(12)*G(13)+G(2)*G(3)*G(7)*G(12)+G(2)*G(
     3)*G(9)*G(13)+G(2)*G(4)*G(5)*G(7)*G(10)*G(11)*G(12)*G(13)+G(2)*G(4
     )*G(5)*G(8)*G(9)*G(11)*G(12)*G(13)+G(2)*G(4)*G(5)*G(8)*G(11)*G(12)
     +G(2)*G(4)*G(5)*G(10)*G(11)*G(13)+G(2)*G(4)*G(6)*G(7)*G(9)*G(11)*G
     (12)*G(13)+G(2)*G(4)*G(6)*G(7)*G(11)*G(12)+G(2)*G(4)*G(6)*G(9)*G(
     11)*G(13)+G(2)*G(4)*G(6)*G(11)+G(2)*G(4)*G(7)*G(10)*G(12)*G(13)+G(
     2)*G(4)*G(8)*G(9)*G(12)*G(13)+G(2)*G(4)*G(8)*G(12)+G(2)*G(4)*G(10)
     *G(13)+G(5)*G(7)*G(9)*G(11)*G(12)*G(13)+G(5)*G(7)*G(11)*G(12)+G(5)
     *G(9)*G(11)*G(13)+G(5)*G(11)+G(7)*G(9)*G(12)*G(13)+G(7)*G(12)+G(9)
     *G(13)+1.0
```

(A

X_{12}/X_1 =

```
THE NUMERATOR
     COSHF(1)*(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*COSHF(3)
     *(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**2  *SINHF(2)*
     SINHF(4)*SQRTF(1)*SQRTF(2)*SQRTF(3)*SQRTF(4)+COSHF(1)*(COSHF(2)*K(
     1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*COSHF(3)*S**3  *SINHF(2)*
     SQRTF(1)*SQRTF(2)*SQRTF(3)*SQRTF(4)+COSHF(1)*COSHF(3)*(COSHF(4)*K(
     3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**3  *SINHF(4)*SQRTF(1)*
     SQRTF(2)*SQRTF(3)*SQRTF(4)+COSHF(1)*COSHF(3)*S**4  *SQRTF(1)*SQRTF
     (2)*SQRTF(3)*SQRTF(4)

THE DENOMINATOR
     COSHF(1)*COSHF(2)*(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))
     *(COSHF(3)*K(2)*(S*T(2)+1.0)*SQRTF(3)+S*SINHF(3))*(COSHF(4)*K(3)*(
     S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S*SINHF(1)*SINHF(3)*SINHF(4)*
     SQRTF(1)**2  *SQRTF(3)*SQRTF(4)+COSHF(1)*COSHF(2)*(COSHF(2)*K(1)*(
     S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*(COSHF(3)*K(2)*(S*T(2)+1.0)*SQRTF
     (3)+S*SINHF(3))*S**2  *SINHF(1)*SINHF(3)*SQRTF(1)**2  *SQRTF(3)*
     SQRTF(4)+COSHF(1)*COSHF(2)*(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*
     SINHF(2))*(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**2  *
     SINHF(1)*SINHF(4)*SQRTF(1)**2  *SQRTF(3)*SQRTF(4)+COSHF(1)*COSHF(2
     )*(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*S**3  *SINHF(1)
     *SQRTF(1)**2  *SQRTF(3)*SQRTF(4)+COSHF(1)*(COSHF(2)*K(1)*(S*T(1)+
     1.0)*SQRTF(2)+S*SINHF(2))*COSHF(3)*(COSHF(3)*K(2)*(S*T(2)+1.0)*
     SQRTF(3)+S*SINHF(3))*(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(
     4))*S*SINHF(1)*SINHF(2)*SINHF(4)*SQRTF(1)**2  *SQRTF(2)*SQRTF(4)+
     COSHF(1)*(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*COSHF(3)
     *(COSHF(3)*K(2)*(S*T(2)+1.0)*SQRTF(3)+S*SINHF(3))*S**2  *SINHF(1)*
     SINHF(2)*SQRTF(1)**2  *SQRTF(2)*SQRTF(4)+COSHF(1)*(COSHF(2)*K(1)*(
     S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*(COSHF(3)*K(2)*(S*T(2)+1.0)*SQRTF
     (3)+S*SINHF(3))*COSHF(4)*(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*
     SINHF(4))*S*SINHF(1)*SINHF(2)*SINHF(3)*SQRTF(1)**2  *SQRTF(2)*
     SQRTF(3)+COSHF(1)*(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))
     *COSHF(4)*(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**2  *
     SINHF(1)*SINHF(2)*SQRTF(1)**2  *SQRTF(2)*SQRTF(3)+COSHF(1)*COSHF(3
     )*(COSHF(3)*K(2)*(S*T(2)+1.0)*SQRTF(3)+S*SINHF(3))*(COSHF(4)*K(3)*
     (S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**2  *SINHF(1)*SINHF(4)*SQRTF(1
     )**2  *SQRTF(2)*SQRTF(4)+COSHF(1)*COSHF(3)*(COSHF(3)*K(2)*(S*T(2)+
     1.0)*SQRTF(3)+S*SINHF(3))*S**3  *SINHF(1)*SQRTF(1)**2  *SQRTF(2)*
```

(A2

THE DENOMINATOR (continued)

```
SQRTF(4)+COSHF(1)*(COSHF(3)*K(2)*(S*T(2)+1.0)*SQRTF(3)+S*SINHF(3))
*COSHF(4)*(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**2  *
SINHF(1)*SINHF(3)*SQRTF(1)**2  *SQRTF(2)*SQRTF(3)+COSHF(1)*COSHF(4
)*(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**3  *SINHF(1)
*SQRTF(1)**2  *SQRTF(2)*SQRTF(3)+(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF
(2)+S*SINHF(2))*(COSHF(3)*K(2)*(S*T(2)+1.0)*SQRTF(3)+S*SINHF(3))*(
COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S*SINHF(1)**2  *
SINHF(2)*SINHF(3)*SINHF(4)*SQRTF(1)*SQRTF(2)*SQRTF(3)*SQRTF(4)+(
COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*(COSHF(3)*K(2)*(S*
T(2)+1.0)*SQRTF(3)+S*SINHF(3))*(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4
)+S*SINHF(4))*S*SINHF(2)*SINHF(3)*SINHF(4)*SQRTF(1)*SQRTF(2)*SQRTF
(3)*SQRTF(4)+(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*(
COSHF(3)*K(2)*(S*T(2)+1.0)*SQRTF(3)+S*SINHF(3))*S**2  *SINHF(1)**
2  *SINHF(2)*SINHF(3)*SQRTF(1)*SQRTF(2)*SQRTF(3)*SQRTF(4)+(COSHF(2
)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*(COSHF(3)*K(2)*(S*T(2)+
1.0)*SQRTF(3)+S*SINHF(3))*S**2  *SINHF(2)*SINHF(3)*SQRTF(1)*SQRTF(
2)*SQRTF(3)*SQRTF(4)+(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(
2))*(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**2  *SINHF(
1)**2  *SINHF(2)*SINHF(4)*SQRTF(1)*SQRTF(2)*SQRTF(3)*SQRTF(4)+(
COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*(COSHF(4)*K(3)*(S*
T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**2  *SINHF(2)*SINHF(4)*SQRTF(1)*
SQRTF(2)*SQRTF(3)*SQRTF(4)+(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*
SINHF(2))*S**3  *SINHF(1)**2  *SINHF(2)*SQRTF(1)*SQRTF(2)*SQRTF(3)
*SQRTF(4)+(COSHF(2)*K(1)*(S*T(1)+1.0)*SQRTF(2)+S*SINHF(2))*S**3  *
SINHF(2)*SQRTF(1)*SQRTF(2)*SQRTF(3)*SQRTF(4)+(COSHF(3)*K(2)*(S*T(2
)+1.0)*SQRTF(3)+S*SINHF(3))*(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S
*SINHF(4))*S**2  *SINHF(1)**2  *SINHF(3)*SINHF(4)*SQRTF(1)*SQRTF(2
)*SQRTF(3)*SQRTF(4)+(COSHF(3)*K(2)*(S*T(2)+1.0)*SQRTF(3)+S*SINHF(3
))*(COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**2  *SINHF(3
)*SINHF(4)*SQRTF(1)*SQRTF(2)*SQRTF(3)*SQRTF(4)+(COSHF(3)*K(2)*(S*T
(2)+1.0)*SQRTF(3)+S*SINHF(3))*S**3  *SINHF(1)**2  *SINHF(3)*SQRTF(
1)*SQRTF(2)*SQRTF(3)*SQRTF(4)+(COSHF(3)*K(2)*(S*T(2)+1.0)*SQRTF(3)
+S*SINHF(3))*S**3  *SINHF(3)*SQRTF(1)*SQRTF(2)*SQRTF(3)*SQRTF(4)+(
COSHF(4)*K(3)*(S*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**3  *SINHF(1)**
2  *SINHF(4)*SQRTF(1)*SQRTF(2)*SQRTF(3)*SQRTF(4)+(COSHF(4)*K(3)*(S
*T(3)+1.0)*SQRTF(4)+S*SINHF(4))*S**3  *SINHF(4)*SQRTF(1)*SQRTF(2)*
SQRTF(3)*SQRTF(4)+S**4  *SINHF(1)**2  *SQRTF(1)*SQRTF(2)*SQRTF(3)*
SQRTF(4)+S**4  *SQRTF(1)*SQRTF(2)*SQRTF(3)*SQRTF(4)
```

(A2)

```
   * * * OUTPUT FROM REDUCE II * * *

DENOM(    1)= -G(3)*G(20)-G(21)+1.0

G(   26)=       -G(3)*G(20)-G(21)
G(   26)=G(   26)/DENOM(    1)

G(   27)=        G(3)*G(8)*G(21)
G(   27)=G(   27)/DENOM(    1)

G(   28)=       -G(3)*G(16)
G(   28)=G(   28)/DENOM(    1)

G(   29)=       -G(3)*G(17)
G(   29)=G(   29)/DENOM(    1)

G(   30)=       -G(3)*G(18)
G(   30)=G(   30)/DENOM(    1)

G(   31)=       -G(3)*G(19)
G(   31)=G(   31)/DENOM(    1)

G(   32)=        G(3)
G(   32)=G(   32)/DENOM(    1)
.    .    .    .    .    .    .    .
.    .    .    .    .    .    .    .
.    .    .    .    .    .    .    .
G(   94)=       -G(90)
G(   94)=G(   94)/DENOM(   17)

G(   95)=       -G(91)
G(   95)=G(   95)/DENOM(   17)                                  (A3)
      TRANSFER FUNCTION X(   7)/X(   1)

             THE NUMERATOR
                 -G(93)

             THE DENOMINATOR
                 -G(1)*G(95)-G(93)

      TRANSFER FUNCTION X(   7)/X(   2)

             THE NUMERATOR
                 G(1)*G(92)

             THE DENOMINATOR
                 -G(1)*G(95)-G(93)

      TRANSFER FUNCTION X(   7)/X(   3)
             .   .   .
             .   .   .
```

TABLE I. - STEP USED BY FORMAC PROGRAM TO SOLVE HEAD RESPONSE

BLOCK DIAGRAM FOR TRANSFER FUNCTION X_1/X_2

Equation	Step				
	0	1	2	3	4
1	$-X_1 + X_2 - X_6 = 0$				$X_6 = -X_1 + X_2$
2	$G_1X_2 - X_3 = 0$			$G_1G_3X_2 - X_1 + G_2G_3X_6 = 0$	$G_1G_3X_2 - X_1 - G_2G_3X_1 + G_2G_3X_2 = 0$
3	$G_2X_6 - X_4 = 0$	$X_4 = G_2X_6$			
4	$X_3 + X_4 - X_5 = 0$	$X_3 + G_2X_6 - X_5 = 0$	$G_3X_3 + G_2G_3X_6 - X_1 = 0$	$X_3 = \frac{X_1 - G_2G_3X_6}{G_3}$	
5	$G_3X_5 - X_1 = 0$		$X_5 = \frac{X_1}{G_3}$		

TABLE II. - NUMERICAL DATA FOR DUCT DYNAMICS

Line	Length		Line inductance per unit length, L_n		Line capacitance per unit length, C_n		Line resistance per unit length,		Resistance, R_m		Resistance times compliance R_mC_m
	in.	cm	$sec^2/in.^3$	sec^2/m^3	in.	m	sec in.^{3}	sec m^3	sec in.^{2}	sec m^2	sec
A	197	5.0	2.73×10^{-5}	1.66	12.38×10^{-5}	31.44×10^{-7}	0	0	---	---	-----
B	72	1.83	13.14	8.018	.94	2.388	↓	↓	Var.	Var.	0.0697
C	75	1.90	6.71	4.094	1.78	4.521	↓	↓	2.61	4.045×10^3	.0212
D	32	.813	6.71	4.094	1.78	4.521	↓	↓	2.61	4.045	.0212

TABLE III. - COMPONENT DATA JET ENGINE INLET CONTROL

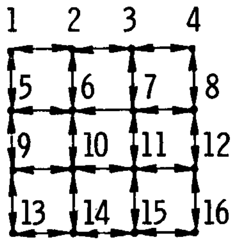
$$G_1 = \frac{846.7\,(-0.833\times10^{-2}\,S^3 + 0.1\,S^2 - 0.5\,S + 1.0)}{S^6 + 18.28\,S^5 + 155.4\,S^4 + 743.9\,S^3 + 2038\,S^2 + 2823\,S + 846.7}$$

$$G_1 = \frac{7.06\,e^{-1.0\,S}}{S^3 + 6.33\,S^2 + 20.01\,S + 7.06} \quad \text{Exact.} \quad G_2 = 846.7, \; G_3 = \frac{1}{S}$$

$G_4, G_5, \ldots, G_9 = 2.041662, -0.3595, -2.54, +8.723, -1.5025, -146.932$

$G_{10}, G_{11}, \ldots, G_{15} = 846.7, 2823, 2038, 743.9, 155.4, 18.28$

$G_{16}, G_{17}, \ldots, G_{21} = 998.9, 600.97, 155.18, 20.745, 1.300, 0.038$



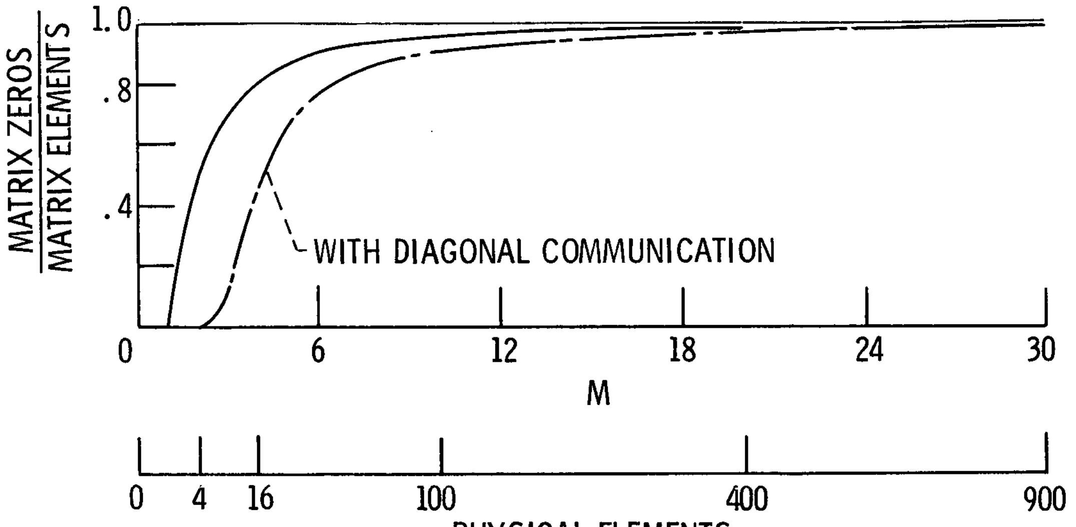
(A) PHYSICAL ARRANGEMENT M = 4, 16 ELEMENTS.

FROM ELEMENT

TO ELEMENT

$$\begin{array}{c|cccccccccccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\
\hline
1 & & 1 & & & 1 & & & & & & & & & & & \\
2 & 1 & & 1 & & & 1 & & & & & & & & & & \\
3 & & 1 & & 1 & & & 1 & & & & & & & & & \\
4 & & & 1 & & & & & 1 & & & & & & & & \\
5 & 1 & & & & & 1 & & & 1 & & & & & & & \\
6 & & 1 & & & 1 & & 1 & & & 1 & & & & & & \\
7 & & & 1 & & & 1 & & 1 & & & 1 & & & & & \\
8 & & & & 1 & & & 1 & & & & & 1 & & & & \\
9 & & & & & 1 & & & & & & & & 1 & & & \\
10 & & & & & & 1 & & & 1 & 1 & & & & 1 & & \\
11 & & & & & & & 1 & & & 1 & 1 & & & & 1 & \\
12 & & & & & & & & 1 & & & 1 & 1 & & & & 1 \\
13 & & & & & & & & & 1 & & & & & & & \\
14 & & & & & & & & & & 1 & & & 1 & 1 & & \\
15 & & & & & & & & & & & 1 & & & 1 & 1 & \\
16 & & & & & & & & & & & & 1 & & & 1 & 1
\end{array}$$

(B) MATRIX SHOWING COMMUNICATION OF ELEMENTS, M = 4.



(C) MATRIX SPARSITY AS FUNCTION OF SYSTEM SIZE.

Figure 1

A-6

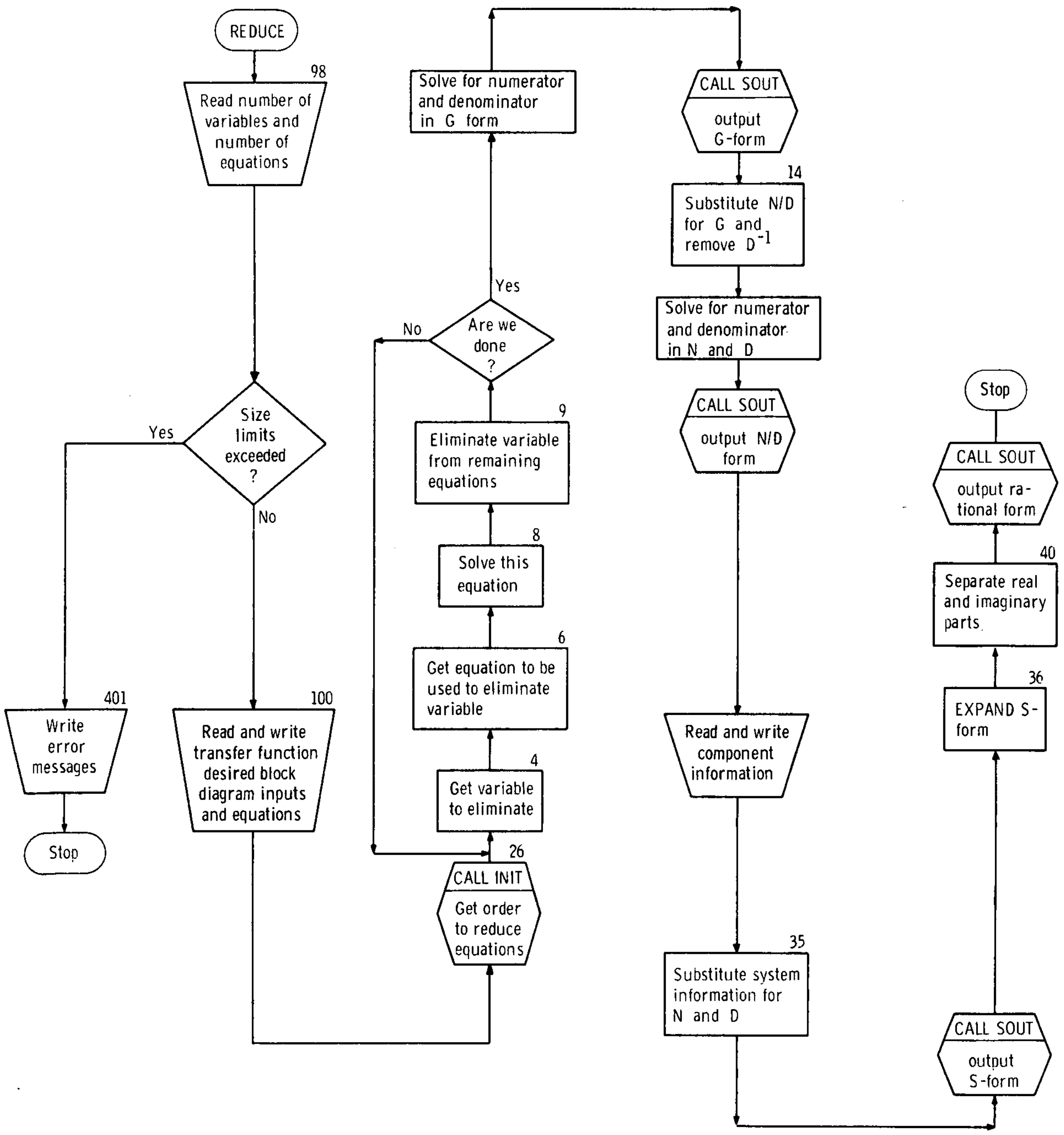
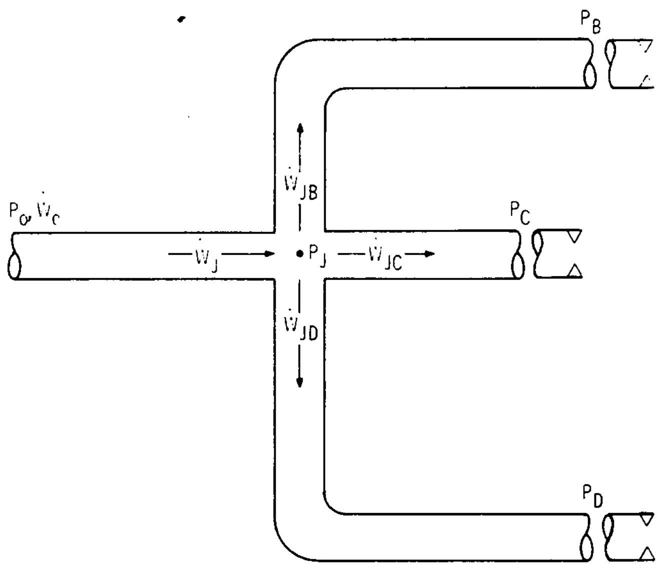
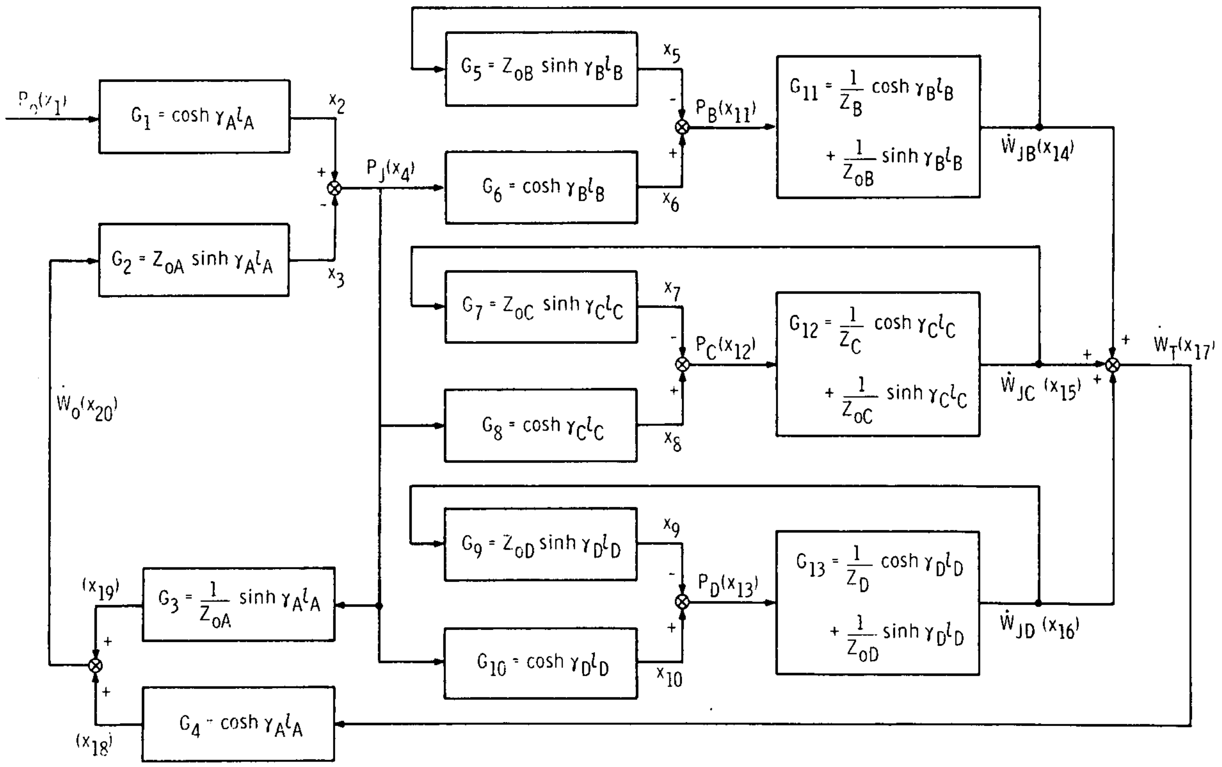

REDUCE
98
Read number of variables and number of equations
Size limits exceeded ?
Yes
No
401
Write error messages
Stop
100
Read and write transfer function desired block diagram inputs and equations
26
CALL INIT
Get order to reduce equations
4
Get variable to eliminate
6
Get equation to be used to eliminate variable
8
Solve this equation
9
Eliminate variable from remaining equations
Are we done ?
No
Yes
Solve for numerator and denominator in G form
CALL SOUT
output G-form
14
Substitute N/D for G and remove D^-1
Solve for numerator and denominator in N and D
CALL SOUT
output N/D form
Read and write component information
35
Substitute system information for N and D
CALL SOUT
output S-form
36
EXPAND S-form
40
Separate real and imaginary parts
CALL SOUT
output rational form
Stop

Figure 2. - Simplified flow chart of REDUCE.

A-7



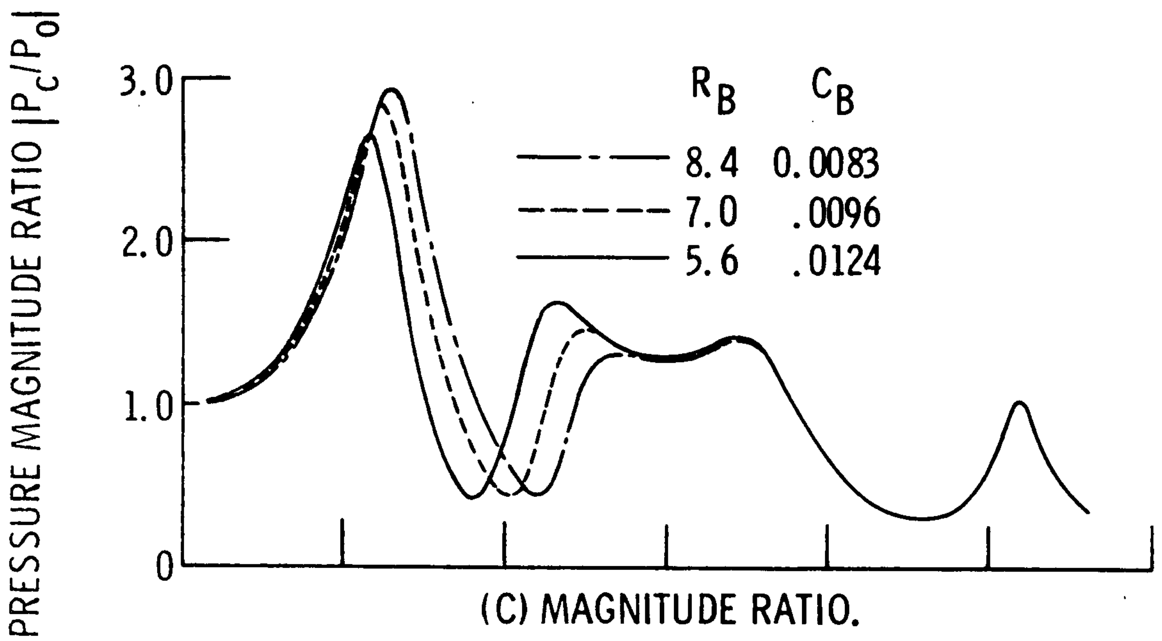
(A) PHYSICAL SYSTEM.



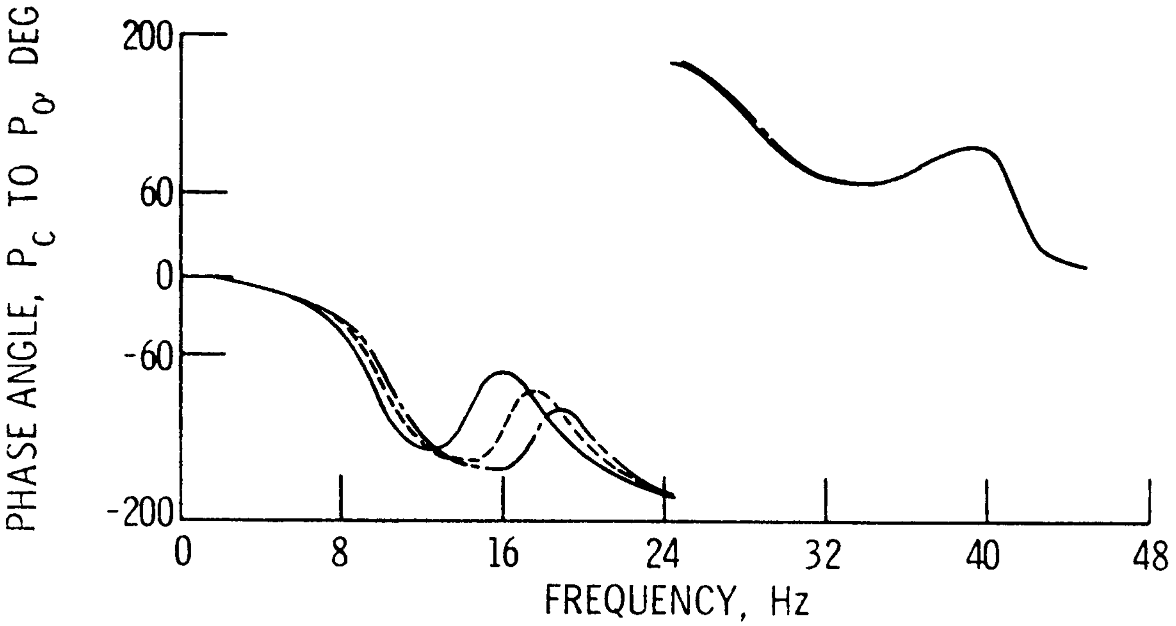
$P_0(x_1)$
$G_1 = \cosh \gamma_A l_A$
$G_2 = Z_{0A} \sinh \gamma_A l_A$
$G_3 = \frac{1}{Z_{0A}} \sinh \gamma_A l_A$
$G_4 = \cosh \gamma_A l_A$
$G_5 = Z_{0B} \sinh \gamma_B l_B$
$G_6 = \cosh \gamma_B l_B$
$G_7 = Z_{0C} \sinh \gamma_C l_C$
$G_8 = \cosh \gamma_C l_C$
$G_9 = Z_{0D} \sinh \gamma_D l_D$
$G_{10} = \cosh \gamma_D l_D$
$G_{11} = \frac{1}{Z_B} \cosh \gamma_B l_B + \frac{1}{Z_{0B}} \sinh \gamma_B l_B$
$G_{12} = \frac{1}{Z_C} \cosh \gamma_C l_C + \frac{1}{Z_{0C}} \sinh \gamma_C l_C$
$G_{13} = \frac{1}{Z_D} \cosh \gamma_D l_D + \frac{1}{Z_{0D}} \sinh \gamma_D l_D$
x_2
x_3
$P_J(x_4)$
x_5
x_6
x_7
x_8
x_9
x_{10}
$P_B(x_{11})$
$P_C(x_{12})$
$P_D(x_{13})$
$\dot{W}_{JB}(x_{14})$
$\dot{W}_{JC}(x_{15})$
$\dot{W}_{JD}(x_{16})$
$\dot{W}_T(x_{17})$
(x_{18})
(x_{19})
$\dot{W}_0(x_{20})$

(B) BLOCK DIAGRAM.

Figure 3. - Line dynamic response.



(C) MAGNITUDE RATIO.



(D) PHASE ANGLE (PRINCIPLE VALUE).

Figure 3. - Numerical solution for line dynamic response.

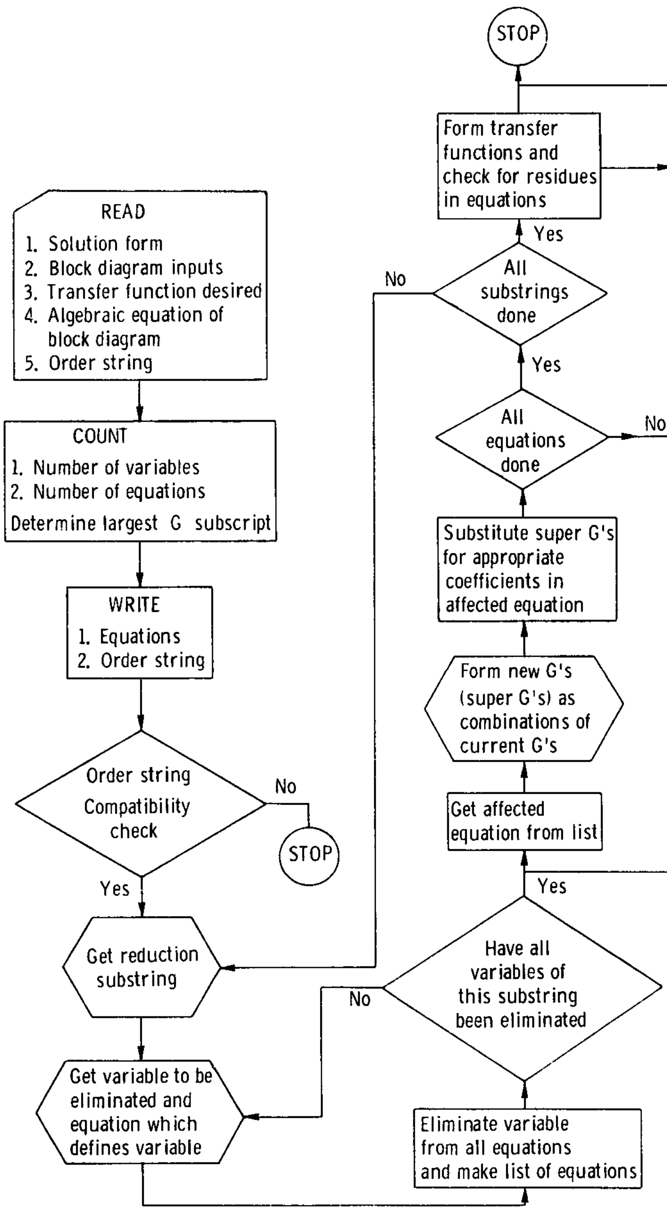

STOP
Form transfer functions and check for residues in equations
Write error
Yes
No
All substrings done
Yes
All equations done
No
Substitute super G's for appropriate coefficients in affected equation
Form new G's (super G's) as combinations of current G's
Get affected equation from list
Yes
Have all variables of this substring been eliminated
No
Eliminate variable from all equations and make list of equations
READ
1. Solution form
2. Block diagram inputs
3. Transfer function desired
4. Algebraic equation of block diagram
5. Order string
COUNT
1. Number of variables
2. Number of equations
Determine largest G subscript
WRITE
1. Equations
2. Order string
Order string Compatibility check
No
STOP
Yes
Get reduction substring
Get variable to be eliminated and equation which defines variable

Figure 4. - Simplified flow chart of REDUCE II program.

A-10

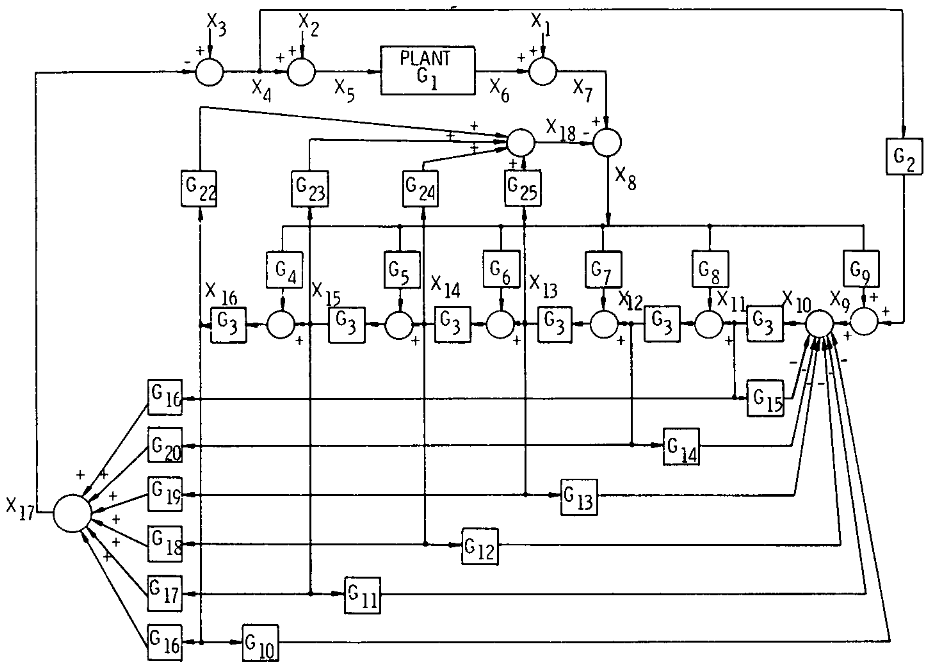


Figure 5. - Block diagram jet engine inlet control.

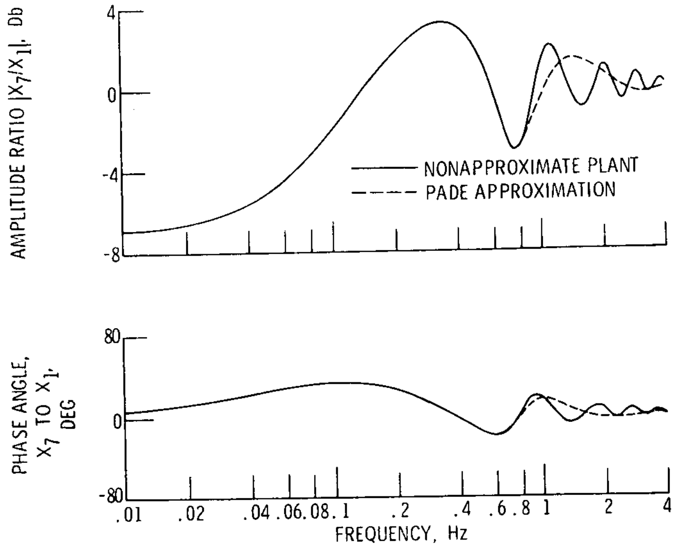


Figure 6. - Typical transfer function results jet engine inlet control.

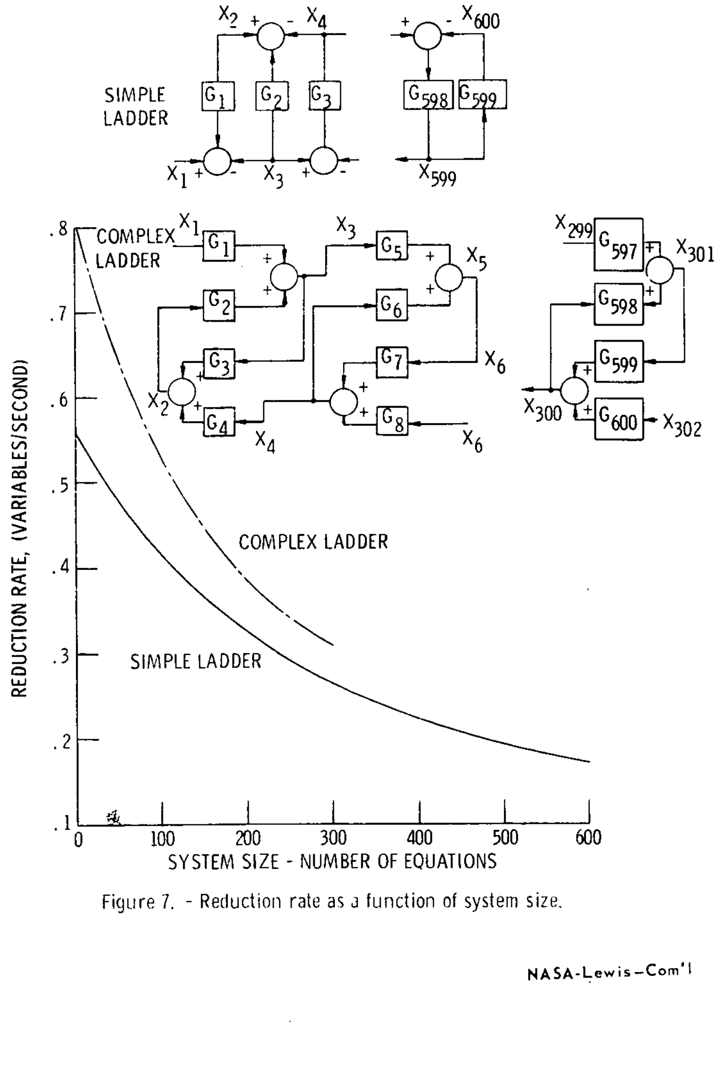


Figure 7. - Reduction rate as a function of system size.