NASA Technical Paper 1818

Reduced Order Feedback Control Equations for Linear Time and Frequency Domain Analysis

Harold P. Frisch

JUNE 1981

NASA

NASA Technical Paper 1818

Reduced Order Feedback Control Equations for Linear Time and Frequency Domain Analysis

Harold P. Frisch
Goddard Space Flight Center
Greenbelt, Maryland

NASA
National Aeronautics
and Space Administration

**Scientific and Technical
Information Branch**

1981

All measurement values are expressed in the International System of Units (SI) in accordance with NASA Policy Directive 2220.4, paragraph 4.

ABSTRACT

An algorithm has been developed and discussed which can be used to obtain reduced order feedback control equations for both time and frequency domain analysis. In a more general context, the algorithm computes a real nonsingular similarity transformation matrix which reduces a real nonsymmetric matrix to block diagonal form, each block of which is a real quasi-upper triangular matrix. The algorithm works with both defective and derogatory matrices and when and if it fails, the resultant output can be used as a guide for the reformulation of the mathematical equations that lead up to the ill-conditioned matrix which could not be block diagonalized.

CONTENTS

REDUCED ORDER FEEDBACK CONTROL EQUATIONS FOR LINEAR TIME AND FREQUENCY DOMAIN ANALYSIS

Harold P. Frisch
Goddard Space Flight Center
Greenbelt, Maryland

INTRODUCTION

The National Aeronautics and Space Administration's (NASA's) large space systems technology (LSST) program is attempting to develop an interactive integrated analysis capability (IAC). Its objective is to provide engineers with an interdisciplinary analysis capability supportive of static, time and frequency domain design and performance evaluation methods. In order to achieve this objective, standard analysis methods must be critically evaluated and their suitability determined for moderate- and large-order system time and frequency domain response analysis.

Frequency response methods for the last several decades have been of fundamental importance in the design and performance evaluation of feedback control systems. Nearly all of the frequency response methods collected in Reference 16 require that the user obtain transfer functions. For single input, single output systems the transfer function is normally expressed as a polynomial ratio in either the Laplace variable s or the Z-transform variable z. For multivariate systems, a matrix of transfer functions must be obtained, each matrix element being definable as a polynomial ratio.

Transfer functions or transfer function matrices can be derived in a variety of ways; e.g., block diagram algebra, Reference 4, chapter 7, or state variable approach, Reference 3, section III-D. Basically, however, once the order of the plant and the order of the feedback control system are defined, the degree of the polynomial ratios used to define the transfer functions are also defined, by default. Since the polynomial ratios are not normally expressible in a truncatable series format, reduced order transfer functions are not generally obtainable, and hence polynomial degree cannot be reduced based upon engineering judgment.

Whenever both plant and controller need be modeled via moderate- to large-order system equations, moderate- to large-degree polynomial ratios will result. These are not only difficult to work with, numerically, but they also tend to place engineering insight behind an opaque curtain of numbers. Since "The purpose of computing is insight, not numbers" (Hamming, Reference 13), a method for circumventing this problem must be developed.

The purpose of this paper is twofold: first, to present a method for obtaining transfer functions expressible as a partial fraction expansion which can be truncated via engineering judgment; and

second, to describe the computer algorithm that does the job. This algorithm is a direct extension of the computer subroutine BDIAG developed by Bavely and Stewart and presented in Reference 2.

Briefly, a linear plant subject to linear feedback control leads via the state variable approach to a set of first order differential or difference equations which have a nonsymmetric coefficient matrix. There is no guarantee that this matrix will have a full set of distinct eigenvalues and furthermore, there is no guarantee that a full set of linearly independent eigenvectors will exist.

In the literature, two methods which lead to a decoupling of the system equations are presented:

- Method one requires the determination of the transformation matrix that will reduce the nonsymmetric coefficient matrix to Jordan canonical form (JCF); this is numerical trouble, to quote Wilkinson, (Reference 9, page 582), "The principle objective of the remainder of this paper is to show the basic limitations of the JCF from the point of view of practical computation, and, indeed to cast doubt on the advisability of trying to determine it." See also Reference 17, page 823, where van Loan unequivocably states that "the JCF cannot be computed using floating point arithmetic."

- Method two requires the determination of the singular value decomposition of the non-symmetric coefficient matrix. This is a well-conditioned operation which can be readily done via subroutine SVD presented in the EISPACK library of eigenanalysis subroutines, References 19, 8, and also Reference 21. In brief, the algorithm computes orthogonal matrices U and V such that the nonsymmetric coefficient matrix A can be expressed as $A = U \Sigma V^T$ where Σ is a diagonal matrix with the non-zero elements along the diagonal being termed "singular values." See Reference 14 for an extensive discussion on the subject. These obtainable results can be used to decouple the system equations; however, the author has been unsuccessful in assigning a physically realizable interpretation to the singular values and associated elements in the matrices U and V. Until this physical interpretation is found, singular values can be used to decouple equations but not to reduce their order via engineering judgment.

In effect, the Bavely and Stewart algorithm avoids the problems associated with the two methods cited above. Jordan canonical forms are not required and the resultant computed data can be physically interpreted. The net result is a real transformation matrix, guaranteed nonsingular, that can be used to "almost" decouple the full set of system equations. Furthermore, the decoupling is with respect to system natural frequencies and associated vectors which can be interpreted as quasi-system modes. This ability to physically interpret results leads to transfer functions expressible as truncatable partial fraction expansions. Truncation decisions are based upon gain coefficients which are real numbers formed by a direct multiplication of a modal observability coefficient and a modal controllability coefficient, along with system (plant + controller) frequencies.

REDUCED ORDER SYSTEM EQUATIONS

Engineers attempting to support the analysis needs of ongoing projects, for the most part, care little about the mathematical details associated with the algorithms used. Are the assumptions and

limitations of the algorithm compatible with analysis needs? Is it easy to use and will it provide results that can be used to enhance engineering insight? These are the questions of fundamental importance to the engineer.

As a consequence, assume that the algorithm BLOCK IT exists (Reference 5). What are its assumptions and limitations, and how can it be used to develop reduced order system equations? These questions are answered in this section. The discussion of the algorithm itself is deferred to the next section.

Consider the set of linear first order ordinary differential equations

$$\dot{X}(t) = A\,X(t) + B\,U(t) \tag{1}$$

and

$$Y(t) = C\,X(t) \tag{2}$$

used to define the dynamics of a linear plant and feedback control system. Recognize also that if the system is either discrete (digital) or sampled data (digital and continuous), then the operative feedback control equations are linear finite difference equations expressible in the form

$$X(k+1) = A\,X(k) + B\,U(k) \tag{3}$$

and

$$Y(k) = C\,X(k) \tag{4}$$

where X = system state variables, U = system input variables, Y = system output variables, A, B and C constant coefficient matrices and (k) implies "state at the time instant t_k." See Reference 6 for the methodology required to set up equations of the form 3 and 4 for sampled data feedback control systems.

It is a well-known fact that if the matrix A in either equation 1 or 3 is real and nonsymmetric, its eigenvalues and associated eigenvectors can be both real and complex. In feedback controls work, the normal situation is that some will be real and the rest will be complex conjugate pairs. Furthermore, if repeated eigenvalues exist, then it is quite possible that A will have less than a full set of independent eigenvectors. In application, this is not an uncommon situation. For example

$$\ddot{X}(t) = 0 \tag{5}$$

expressed as

$$\begin{Bmatrix} \dot{X}(t) \\ \ddot{X}(t) \end{Bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{Bmatrix} X(t) \\ \dot{X}(t) \end{Bmatrix} \tag{6}$$

has an "A" matrix with two eigenvalues equal to zero and only one linearly independent eigenvector. In matrix theory terminology, the matrix A may be both defective and derogatory (see (Reference 11 and Reference 18, chapter III).

Accept for the moment that a well-conditioned (with respect to matrix inversion) real, nonsingular similarity transformation matrix ϕ can be numerically computed, and then used to reduce the matrix A to block diagonal form. That is

$$\phi^{-1} A \phi = \text{diag} [G_1, G_2, ..., G_k] \tag{7}$$

where each matrix partition G_i, i=1,2,...,k is real quasi-upper triangular. In application, most diagonal blocks G_i will be either order one, a real eigenvalue or order 2, a complex conjugate pair. Only those eigenvalues that either do not have an associated eigenvector or have one that is nearly parallel to another are grouped in blocks of order greater than 2. The algorithm is set up to search for nearly parallel quasi-eigenvectors and to group their associated eigenvalues together into a single block. The net effect of this is to group potential numerical problems together and then to stop the reduction process just before numerical computation problems start.

It is important to note that since each block is in quasi-upper triangular form, each block can be associated with a subset of the eigenvalues of A; furthermore each eigenvalue in each subset is computable via a simple algebraic equation that utilizes only those real numbers on the diagonal and adjacent upper and lower subdiagonals (Reference 21, contribution II/14). This is the central point of the algorithm. It is this point which makes physical interpretation of computational results possible, and leads on to the ability to obtain reduced order equations.

The algorithm also has the highly desirable feature that it "won't" always work. If the analyst attempts to mathematically model a well-conditioned physical system with an ill-conditioned computing problem, the algorithm will fail to compute the desired low-order blocks. It is impossible for any algorithm to mathematically recreate critical information that the analyst has destroyed by a poor choice of analytical procedures (Reference 7). This point will be explored later in this report, since data associated with the failure yields information that can be utilized as a guide for restructuring the analytical procedures which were the source of the computational problems.

Within the algorithm, a posteriori error analysis is used in various logic loops. It is also used as a stopping criterion. Let ϕ, ϕ^{-1} and G_i, i=1,2,...,k be the actual matrices computed and let A be the input matrix of order N, known exactly. The reduction process stops when

$$\left\| \phi \, \text{diag} [G_1, G_2, ..., G_k] \phi^{-1} - A \right\| \leq 10 * N^P * \left\| A \right\| * 16^{-13} \tag{8}$$

where $\|\cdot\|$ implies square root of the sum of the squares and 16^{-13} is the machine precision constant for IBM-360 double precision computation which uses 13, hexadecimal (base 16) digits to define the matissa of each floating point number. The number P is a user-controllable parameter whose default value is set to 1.75. This default value is slightly more stringent than what is guaranteed when one computes all eigenvalues via orthogonal transformations and the QR-algorithm, (Reference 21, pp. 353 and 367; Reference 7).

To proceed, define q as the set of generalized time dependent system coordinates, related to the set of system state variables by the linear transformation equation

$$X(k) = \phi\, q(k) \tag{9}$$

The analogy between eigenvectors and generalized displacement coordinates in structural analysis is intentional and should be apparent to the reader. Substitute directly into equations 3 and 4, use equation 7 and obtain

$$q(k+1) = \text{diag}[G_1, G_2, \ldots, G_k]\; q(k) + \phi^{-1} B U(k) \tag{10}$$

and

$$Y(k) = C\phi\, q(k) \tag{11}$$

From equation 10 it should be noted that the complete set of system equations has been transformed into a set of "almost" decoupled system equations. By truncating out those generalized system coordinates that do not effect output variable response, as defined by equation 11, reduced order time domain equations can be obtained.

If frequency domain analysis is the objective, transform equations 3 and 4 via the Z-transform, use equation 7, rearrange, combine and obtain

$$Y(z) = C\phi\, \text{diag}\,[(z\mathbb{1}-G_1)^{-1}, (z\mathbb{1}-G_2)^{-1}, \ldots, (z\mathbb{1}-G_k)^{-1}]\, \phi^{-1} B U(z) \tag{12}$$

where $\mathbb{1}$ is a unit diagonal matrix of appropriate order.

In application, the rectangular matrix C is usually a user-defined quantity that normally is extremely sparse; i.e., each row will usually contain only one or two non-zero real numbers, usually ± 1.

The matrix product $C\phi$ can be physically interpreted as a "system mode observability matrix." In particular the row partition

$$\mathscr{C}_{kj} = \sum_i C_{ki}\phi_{ij} \tag{13}$$

defines the degree to which the system modes associated with block j can be observed by output state variable k.

The elements of the rectangular matrix B are for complex problems not as readily definable as those of C. If the Dynamic Interaction Simulation of Controls and Structure (DISCOS) program, Reference 3, is used for the derivation of linearized plant equations, and the methodology of Reference 6 is used to set up equations of the form 1 and 2 or 3 and 4, the elements of B are computable. Each column of the matrix B defines the degree to which the associated input signal drives each system state variable. The matrix product ϕ^{-1} B can be physically interpreted as a "system mode controllability matrix." In particular the row partition

$$\mathcal{O}_{jm} = \sum_{i} \phi_{ji}^{-1} B_{im} \tag{14}$$

defines the degree to which the system modes associated with block j can be controlled by input variable m.

Reduced order transfer functions are obtained by direct implementation of the fact that for the system modes associated with block j to contribute to the transfer function between output state variable Y_k and input variable U_m, they must be both observable to Y_k and controllable by U_m.

Direct utilization by equations 13 and 14 leads to the equation for the transfer function mentioned, that is

$$Y_k(z) = \left[\sum_{j} \mathcal{C}_{kj} (z \mathbb{1} - G_j)^{-1} \mathcal{O}_{jm} \right] U_m(z) \tag{15}$$

where it is reiterated that all elements of the system mode controllability and observability matrices are real numbers. If standard eigenanalysis procedures were employed to more completely decouple rather than to just "almost" decouple the equations, the analogous numbers would have been complex and not readily interpretable.

It should be reasonably evident from equation 15 that first, second, third and higher order contributors to the transfer function can be studied separately or in combination by selectively deleting terms in the summation defined by equation 15. This is ideal for the generation of engineering insight since one can truncate all but primary contributors first, and then successively add less significant contributors one by one.

In application, an implementation problem exists when the order of G_j is greater than 2. When this occurs, the full matrix $(z\,\mathbb{1} - G_j)^{-1}$ must be computed; when the order is 2, one can simply use the fact that

$$\begin{bmatrix} z - g_{11} & -g_{12} \\ -g_{21} & z - g_{22} \end{bmatrix}^{-1} = \frac{\begin{bmatrix} z - g_{22} & g_{12} \\ g_{21} & z - g_{11} \end{bmatrix}}{[z^2 - (g_{11} + g_{22})z + g_{11}g_{22} - g_{12}g_{21}]} \tag{16}$$

and when order is greater than 2, the full matrix $(z\,\mathbb{1} - G_j)^{-1}$ is computable via the Method of Leverrier defined and discussed in Reference 7. It is important to recognize that this method is not practical for computation with large-order matrices. However, in application, if the computing problem is well conditioned, then one is virtually guaranteed that all G_j will be of an order low enough for Leverrier's method to be applicable.

The next implementation detail which must be addressed is that after each required matrix $(z\,\mathbb{1} - G_j)^{-1}$ is evaluated and substituted into equation 15, the net result will be a transfer function expressed as a partial fraction expansion of the form

$$Y_k(z) = \left[\sum_j \frac{N_j(z)}{D_j(z)}\right] U_m(z) \tag{17}$$

Poles and zeros for this transfer function must be computed. The poles are simply the system eigenvalues that were not truncated out. The zeros are computable via a straight forward method presented in Reference 7 that makes use of the fact that the zeros of the partial fraction expansion are the roots of the determinantal equation

$$\det \begin{bmatrix} 0 & D_2(z) & D_k(z) & 1 \\ D_1(z) & 0 & D_k(z) & 1 \\ D_1(z) & D_2(z) & \cdot & \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \\ D_1(z) & D_2(z) & 0 & 1 \\ N_1(z) & N_2(z) & N_k(z) & 0 \end{bmatrix} = 0 \tag{18}$$

Another application of BLOCK IT is in the evaluation of the matrix exponential. From method 18, Reference 17 and equation 7 herein

$$e^{At} = \phi^{-1} \text{ diag}[e^{G_1 t}, e^{G_2 t}, \ldots, e^{G_k t}]\phi \tag{19}$$

BLOCK IT

The program BLOCK IT, Reference 5, is the outgrowth of an attempt by the author to more fully understand and utilize an algorithm for computing reducing subspaces by block diagonalization, i.e., subroutine BDIAG, Reference 2. BLOCK IT should be viewed as an extension of BDIAG. It is specifically tailored to meet the needs of the analyst requiring reduced order dynamic models for feedback controls analysis. It contains additions and alterations to the original code that are useful in overcoming the computational problems that limited the usefulness of the original code in controls analysis application.

The purpose of the program BLOCK IT is to call the sequence of subroutines required to compute the transformation matrix ϕ, its inverse ϕ^{-1} and each of the diagonal blocks Gi, i = 1, 2, . . . , k appearing in equation 7. The program has been designed so that the user can use it either in a default mode or in a mode that permits user interface. By utilization of the interface capability, the user is given the ability to iteratively determine the maximum degree of decoupling that can be achieved while still keeping computational error within the error bounds dictated by equation 8.

The primary difference between BLOCK IT and BDIAG is in the logic used to select which of the system eigenvalues should be blocked together when complete decoupling fails. Complete decoupling occurs when each diagonal block can be associated with either a particular real eigenvalue or a particular complex pair of eigenvalues, and when the associated transformation matrix ϕ, its inverse ϕ^{-1} and the blocks themselves can be used to reproduce the original matrix A to a precision defined by equation 8. In addition, many other alterations to BDIAG have been introduced that are intended to both increase and check the accuracy of the resultant computation.

BLOCK IT, unlike BDIAG, is a multistep iterative operation. It does, however, have a computation stop built in—if blocking is unsuccessful after three attempts. In BDIAG, the user defined a constant that was intended to be a measure of desired computational precision; this was then used to block diagonalize the matrix A in one step. It was found after an extensive series of tests, with several of the matrices defined in Reference 12, chapter 5, that the user-defined number simply did not do its intended job. As a consequence, new blocking logic has been developed and incorporated into BLOCK IT.

The logic makes use of the fact that the inability to decouple is due to the near-singular nature of the computed transformation matrix ϕ, rather than to a buildup of computational errors in the elements of ϕ. The near-singular nature of ϕ is a reflection of the fact that a full set of independent

eigenvectors do not exist for the problem at hand. It should be noted that if one attempts to compute all eigenvectors for a matrix with less than a full set of eigenvectors via the EISPAC algorithm HQR2 (Reference 19, page 338), the algorithm will run to completion, without an error stop, and yield an eigenvector matrix with near-parallel eigenvectors, (Reference 21, page 389). Utilization of the fact that computed eigenvectors for all eigenvalues in a Jordan block are nearly parallel is used to define the blocking logic in BLOCK IT. This allows the algorithm to handle defective and derogatory matrices.

In application, repeated eigenvalues are not the only source of near-parallel eigenvectors. The other prime source is the analyst's use of poor mathematical modeling techniques. For example, use of absolute displacement coordinates rather than relative displacement coordinates or vice versa, use of a Euler angle sequence near gimbal lock, inclusion of negligible time delays, . . . , all lead to near parallel eigenvectors with associated eigenvalues that may be widely separated in magnitude.

Another, although not so obvious, poor modeling technique that can lead to numerical problems is the inclusion of explicit high degree characteristic polynomials in the definition of a feedback control system. For example, consider the characteristic polynomial given in power series form as

$$S^N + a_{N-1} S^{N-1} + \ldots + a_1 S + a_0 = 0 \tag{20}$$

and its companion matrix

$$A = \begin{bmatrix} 0 & 1 & & & & & & \\ 0 & 0 & 1 & & & & & \\ . & . & . & & & & & \\ . & . & . & & & & & \\ 0 & 0 & 0 & . & . & . & 0 & 1 \\ -a_0 & -a_1 & -a_2 & . & . & . & -a_{N-1} & -a_N \end{bmatrix} \tag{21}$$

This is a matrix in Frobenius form.

If the roots of equation 20 are distinct and are given by $\lambda_1, \lambda_2, \ldots, \lambda_N$ then it is well known (Reference 10, page 245 or Reference 15, page 493) that the similarity transformation that reduces A to diagonal form is the Vandermonde matrix V, where

$$V = \begin{bmatrix} 1 & 1 & 1 \\ \lambda_1 & \lambda_2 & \lambda_N \\ \lambda_1^2 & \lambda_2^2 & \lambda_N^2 \\ . & . & . \\ . & . & . \\ . & . & . \\ \lambda_1^{N-1} & \lambda_2^{N-1} & \lambda_N^{N-1} \end{bmatrix} \tag{22}$$

That is

$$V^{-1} A V = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N) \tag{23}$$

where column i of the Vandermonde matrix is the eigenvector associated with eigenvalue λ_i. From equation 22, it should be reasonably apparent that eigenvalues that are clearly distinct may have eigenvectors that are nearly parallel, and as N increases, the problem of near-parallelism becomes decidedly worse.

Suppose that equation 20 stemmed from the set of simultaneous first order differential equations

$$\begin{aligned} \dot{X}_1 \quad &+ \lambda_1 X_1 \quad &&= X_2 \\ &\quad . \\ &\quad . \\ &\quad . \\ \dot{X}_{N-1} \quad &+ \lambda_{N-1} X_{N-1} \quad &&= X_N \\ \dot{X}_N \quad &+ \lambda_N X_N \quad &&= 0 \end{aligned} \tag{24}$$

The characteristic matrix A* associated with the set of Laplace transform equations derived from equation 24 is

$$A^* = \begin{bmatrix} \lambda_1 & -1 & & \\ & \lambda_2 & -1 & 0 \\ & & . & . \\ & & . & . \\ & & . & . \\ & 0 & \lambda_{N-1} & -1 \\ & & & \lambda_N \end{bmatrix} \tag{25}$$

This equation is upper triangular and hence the eigenvalues are given exactly as the diagonal elements $\lambda_1, \lambda_2, \ldots, \lambda_N$. Furthermore it can be shown that the similarity transformation matrix which reduces A* to diagonal form is the upper triangular matrix

$$V^* = \begin{bmatrix} 1 & \frac{1}{\lambda_1 - \lambda_2} & \frac{1}{(\lambda_1 - \lambda_2)(\lambda_2 - \lambda_3)} & \cdots & \frac{1}{(\lambda_1 - \lambda_N)(\lambda_2 - \lambda_N)\ldots(\lambda_{N-1} - \lambda_N)} \\ & 1 & \frac{1}{(\lambda_2 - \lambda_3)} & & \frac{1}{(\lambda_2 - \lambda_N)(\lambda_3 - \lambda_N)\ldots(\lambda_{N-1} - \lambda_N)} \\ & & \ddots & & \vdots \\ & & & & \frac{1}{(\lambda_{N-1} - \lambda_N)} \\ & & & & 1 \end{bmatrix} \quad (26)$$

That is

$$V^{*-1}\ A^*\ V^* = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N) \quad (27)$$

where column i of V* is the eigenvector associated with the eigenvalue λ_i. From equation 26, it should be reasonably apparent that in general eigenvectors will not be parallel. The important point is that there is no tendency toward near-parallelism as N increases.

The key to realizing what has happened is in the recognition that $X_1, X_2, \ldots, X_N$ in equation 24 are the state variables that have physical meaning, i.e., they are usually variables that are measurable in a true physical sense. These are retained in the development path leading to equations 26 and 27. On the other hand, the path going over to equation 20 via transform algebra introduces a new set of state variables, namely X_1 and its first N-1 time derivatives. These are difficult to measure physically and are also difficult to work with numerically.

To proceed, view each column of the computed transformation matrix ϕ, of equation 8, as a quasi-eigenvector that can be associated with a particular system eigenvalue. From Reference 22, page 55, compute the generalized angle θ_{ij} between the vector $\vec{\phi}_i$ defined by column i of ϕ and the vector $\vec{\phi}_j$ defined by column j of ϕ, via the equation

$$\theta_{i,j} = \cos^{-1}\left[\frac{\vec{\phi}_i \cdot \vec{\phi}_j}{|\vec{\phi}_i| \quad |\vec{\phi}_j|}\right] \quad (28)$$

The elements in the matrix Θ composed of all $\theta_{i,j}$ can be used to set up a blocking sequence for the next attempt at blocking. If the user selects the default option, all eigenvalues that have associated quasi-eigenvectors within 12.5 degrees are grouped together before an attempt to block is made. The net effect is to gather potential numerical problems together without making a futile attempt at computing the impossible. If this fails, 2 * 12.5 degrees is used on the next attempt and then upon failure, 3 * 12.5 degrees. In application, blocking is normally successful with 12.5°. If not, it is generally better to reset the 12.5° to better conform to the data provided by the printout of the contents of the array Θ, rather than to increase the number of iterative cycles. The choice of 12.5° has no theoretical basis, it simply works well in application.

In order to present a definition of BLOCK IT in as clear a manner as possible, the following discussion parallels the sequence of computational steps carried out by the algorithm. The intention is to provide the rationale and purpose of each step and to rely on referenced literature to provide in-depth details.

The algorithm is basically a continuation of the procedure used to compute eigenvalues. That is, a sequence of similarity transformations are computed that keep reducing the partially reduced matrix to successively simpler forms. The sequential procedure ends when the reduced form of the matrix A is block diagonal. The resultant similarity transformation that can be used to reduce A directly to block diagonal form is the accumulated product of the entire sequence of similarity transformations used in the reduction process.

Step 1. Use EISPACK subroutine BALANC, Reference19, page 200 and Reference 21, page 315 to balance the input matrix A. This step is not in BDIAG. It is included in BLOCK IT to insure that eigenvalues that can be isolated via zero error row column permutations are so isolated and are not recomputed. BALANC is also used to reduce the norm of A in such a manner that eigenvalues can be computed with improved accuracy.

Step 2. Use EISPACK subroutine ORTHES, Reference 19, page 406 and Reference 21, page 339 to reduce the real general matrix A to upper Hessenberg form using orthogonal similarity transformations. This step is in BDIAG. It is chosen for use because of its guaranteed bound on numerical round off error.

Step 3. Use EISPACK subroutine ORTRAN, Reference 19, page 412. This step accumulates the transformations that were used in ORTHES.

Step 4. Reduce the matrix in upper Hessenberg form to quasi-triangular form via the QR algorithm, and accumulate the similarity transformations used. This is done in subroutine HQR2BC, which is an adaptation of EISPACK subroutine HQR2, Reference 19, page 338 and Reference 21, page 359. HQR2BC is a duplication of HQR2, with the step to compute eigenvectors deleted and the code required to compute a few arrays needed for follow-on computation added. This is a deviation from BDIAG. Reason: for the series of test problems used, subroutine HQR2BC computed eigenvalues, at or in the vicinity of zero, more accurately than subroutine HQR3 used in BDIAG.

Step 5. Sweep the computed arrays of numerical garbage, then scan the swept quasi-upper triangular matrix to determine if any blocks can be isolated by zero error row-column permutations. This work is done in the subroutine QUKBLK which was written expressly for BLOCK IT. The need for QUKBLK arose from application. Frequently in large-order systems a certain degree of decoupling may exist; e.g., the roll and yaw control systems may be coupled only when nonzero products of inertia are accounted for. QUKBLK provides the ability to recognize this decoupling prior to the main computation cycle. It then sets up data which is subsequently used to solve the problem as a series of smaller subproblems rather than one large problem. The net result is an enhancement of numerical precision.

With respect to the sweeping out of numerical garbage, the validity of this step can be argued. The author counterargues that in application it normally helps more than it hurts and at times it is essential. As a consequence, users of BLOCK IT are given the option to sweep or not to sweep. If sweeping is used, the logic is that if the array element plus the norm of the array equals the norm of the array, set the array element to zero. The particular norm used is array dependent; it is chosen to add conservativeness into the sweeping step. The arrays swept are the eigenvalue array, the quasi-upper triangular matrix and the accumulated transformation matrix.

Step 6. Reduce the quasi-upper triangular matrix to block diagonal form and accumulate the transformation matrices. The logic used in this step depends upon whether or not a previous attempt at blocking has been made.

First Attempt. Starting at the upper left hand corner of the matrix, attempt to compute the rectangular matrix P which satisfies the following equation

$$\begin{bmatrix} \mathbb{1} & -P \\ 0 & \mathbb{1} \end{bmatrix} \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} \mathbb{1} & P \\ 0 & \mathbb{1} \end{bmatrix} = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} \tag{29}$$

On the first try A_{11} is either 1 X 1 or 2 X2 and A_{22} is the rest of the quasi-upper triangular matrix being reduced. The problem of determining P, and hence the similarity transformation matrix, then becomes one of solving the equation

$$A_{11}P - P A_{22} = -A_{12} \tag{30}$$

This is a classic matrix equation which is numerically solvable via the algorithm SHRSLV presented in Reference 1 and Reference 2. BLOCK IT uses a variation of SHRSLV as used in BDIAG. The procedure is sequential and involves solving a series of subproblems of the form

$$AX - XB = C \tag{31}$$

where the order of the system of equations defined by equation 31 is at most four. In application, problems were encountered when the elements of the matrix C were all zero; for this case a unique solution to equation 31 does not exist. For BLOCK IT, it is proper to choose and utilize the particular solution X = 0; this is done.

Furthermore, in Reference 2, page 361, it is argued that if all elements of the solution matrix P are less than a user-defined number "RMAX," then the similarity transformation matrix of equation 29 will be well-conditioned with respect to matrix inversion and in application, so should the accumulated product of all similarity transformations. This argument simply did not hold true for the test problems studied by the author. In BLOCK IT the user does not define the number RMAX; as a consequence, the version of SHRSLV in BLOCK IT has all tests with RMAX deleted and now conforms more to the version of SHRSLV published in Reference 1. The net result is that any computable solution is accepted. From Reference 1, "it is well-known that AX + XB = C has a unique solution if and only if the eigenvalues $\alpha_1, \alpha_2, \ldots, \alpha_m$ of A and $\beta_1, \beta_2, \ldots, \beta_n$ of B satisfy $\alpha_i + \beta_j \neq 0$ (i = 1, 2, . . . , m; j = 1, 2, . . . , n)." The fact that the resultant similarity transformation can be highly ill-conditioned with respect to inversion is acceptable to BLOCK IT logic.

Failure to compute a unique solution is still a real possibility. This always occurs when both A_{11} and A_{22} in equation 29 contain eigenvalues identically equal to zero. When this happens, the general procedure is to scan the 1 X 1 and 2 X 2 blocks along the diagonal of A_{22} to find the eigenvalue most equal in magnitude to the average magnitude of the eigenvalues in A_{11}. When found, a series of similarity transformations are computed which move the block containing the sought-after eigenvalue up to the top left corner of A_{22}. A_{11} is then increased in order by 1 or 2 and A_{22} accordingly reduced in order. The process of finding P in equation 29 is then repeated until a matrix P is computed. Upon successful solution, the first 1 X 1 or 2 X2 block in A_{22} is viewed as A_{11} in equation 29 and the process is repeated until A_{22} is only 1 X 1 or 2 X 2. At this point, the block diagonal procedure is complete, some cleanup work is done, and control transferred over to the error diagnostic routine BWRITE, which will be discussed later.

The procedure for moving blocks along the diagonal of a quasi-upper triangular matrix is carried out by subroutine EXCHNG. In BLOCK IT subroutine EXCHNG is a variation of the subroutine EXCHNG defined in Reference 20 and given in Reference 2. In application, it was found that the movement of diagonal blocks containing the eigenvalue zero presented numerical accuracy problems in that the exact zero was destroyed and replaced by a very small number in the exchange process. Furthermore, it was noted that under certain readily testable conditions the exchange could be carried out by a zero error row/column interchange rather than via the computational process. Fixes to both of these problems are incorporated in the BLOCK IT version of EXCHNG.

Not First Attempt. The diagnostic and print data routine BWRITE is used to define the blocking sequence which will group together all eigenvalues having near parallel quasi-eigenvectors. The important feature of this logic is that the grouping is based upon near parallelism

of quasi-eigenvectors and not the nearness of eigenvalues. The implementation of this blocking is carried out by subroutine EXCHNG. It is used to move the preselected 1 X 1 and 2 X 2 blocks of the quasi-upper triangular matrix up to the top left corner. These define the A_{11} matrix in equation 29 which may now be greater than 1 X 1 or 2 X 2. The order of A_{11} is dictated by the blocking directions as computed via BWRITE. Once A_{11} is formed, the associated P matrix of equation 29 is computed. A_{22} is then partitioned into the form of equation 29 via movement of the next group of blocks to its top left corner and the process repeated until A_{22} is either of order 1 or 2. At this point cleanup work is done and control again passed to the error diagnostic subroutine BWRITE.

Step 7. Determine if the computed block diagonal matrix and associated transformation matrix are accurate enough to satisfy the user-specified error criterion defined by equation 8.

If the error criterion is satisfied, the computational process is complete. If the error criterion is not satisfied and more than three unsuccessful tries have been made, the computational process stops with a failure message. Making more than three attempts is usually futile. The most effective procedure is to reset the angular measure that defines "nearness." Normally this is not a very sensitive parameter and it conforms with one's intuitive feeling for nearness when angles are measured on a scale from 0 to 90°.

If the error criterion is not satisfied and less than three attempts have been made, in addition to outputting data, BWRITE also sets up the data required to define the blocking sequence which should be used in the next try. The algorithm simply computes the generalized angle between all pairs of quasi-eigenvectors, columns of the accumulated transformation matrix ϕ. With the array of generalized angles, it is a routine task to scan it with the intent of determining which eigenvalues have near parallel quasi-eigenvectors. Once done, control data can be set up in a format compatible with the needs of step 6 discussed above.

ALGORITHM FAILURE

The tendency for the algorithm to fail to satisfactorily block diagonalize is an attribute. Algorithm failure inevitably stems from the fact that the matrix is ill-conditioned with respect to blocking, and this stems from the fact that too many system eigenvectors are nearly parallel. The root cause of this problem is usually with the analyst who derived the equations which lead to the ill-conditioned matrix. Once one recognizes that a poor choice of system state variables at the equation derivation step is the root of the problem, one is in a position to utilize blocking failure data to restructure the equations so that the resultant computing problem will be well-conditioned. One simply has to scan the matrix of generalized angles θ_{ij} to determine which quasi-system eigenvectors are nearly parallel, note their associated eigen frequencies and then backtrack to determine which system state variables contribute most significantly to system response at these frequencies. These state variables are usually poorly chosen (see Reference 7). A few examples of poor engineering judgment are:

- An attempt to model system response frequencies which differ by many orders of magnitude;

- A poor choice of physical units. Ideally, all numbers of significance in the computing problem should be of comparable order; and

- A poor choice of state variables such as, for example, using a Euler angle sequence which is subject to "gimbal lock," using relative rather than absolute coordinates for rate and/or displacement measures or vice versa.

In application, if blocking fails, stop. Alternate computing methods normally will not help. The computing problem itself is ill-conditioned. Computational methods cannot recreate data which has been destroyed by poor engineering judgment in the equation development step.

TEST CASE

A particularly difficult test problem is presented in Reference 12, problem 5.27. This matrix given by

$$A = \begin{bmatrix} 1 & 1 & 1 & -2 & 1 & -1 & 2 & -2 & 4 & -3 \\ -1 & 2 & 3 & -4 & 2 & -2 & 4 & -4 & 8 & -6 \\ -1 & 0 & 5 & -5 & 3 & -3 & 6 & -6 & 12 & -9 \\ -1 & 0 & 3 & -4 & 4 & -4 & 8 & -8 & 16 & -12 \\ -1 & 0 & 3 & -6 & 5 & -4 & 10 & -10 & 20 & -15 \\ -1 & 0 & 3 & -6 & 2 & -2 & 12 & -12 & 24 & -18 \\ -1 & 0 & 3 & -6 & 2 & -5 & 15 & -13 & 28 & -21 \\ -1 & 0 & 3 & -6 & 2 & -5 & 12 & -11 & 32 & -24 \\ -1 & 0 & 3 & -6 & 2 & -5 & 12 & -14 & 37 & -26 \\ -1 & 0 & 3 & -6 & 2 & -5 & 12 & -14 & 36 & -25 \end{bmatrix} \quad (32)$$

is both defective and derogatory. To be specific, eigenvalue $\lambda = 2.0$ is of multiplicity 5. It is associated with two nonlinear elementary divisors, one of degree 3 and one of degree 2; this implies only two independent eigenvectors for the 5 eigenvalues equal to 2.0. Likewise, eigenvalue $\lambda = 3$ is of multiplicity 4. It is associated with two quadratic elementary divisors; this implies only two independent eigenvectors for the 4 eigenvalues equal to 3.0. To conclude, eigenvalue $\lambda = 1.0$ is of multiplicity 1 and has an independent eigenvector.

The problem has been chosen to demonstrate that BLOCK IT is capable of creating distinct blocks that contain equal eigenvalues, unlike BDIAG (Reference 2) which groups all equal eigenvalues together in a single block. This feature is extremely important in application. For example, if the same control law is used for roll and yaw control, equal eigenvalues can be obtained, but one would be associated with roll control and the other with the yaw control. If the algorithm blocks the two together, the physical decoupling that exists in the actual system is destroyed by the mathematics. BLOCK IT is designed to avoid this annoying problem.

Direct utilization of BLOCK IT, with the angular measure that defines nearness set to 5°, yields the following results which can be used to reproduce the matrix A such that

$$\frac{\|\phi^{-1}\ \mathrm{diag}(G_1, G_2, \ldots, G_5)\phi - A\|}{\|A\|} = .82618*10^{-14} \tag{33}$$

and the condition number of ϕ with respect to inversion is computed to be 671.62. Of course, actual computation of the above numbers was carried out with ϕ, ϕ^{-1} and G_i i = 1, . . . , 5 given to full double precision accuracy. The computed matrices were

$$\phi = \begin{bmatrix}
-.3332 & -.9428 & -.001757 & -.9355 & -.9870 & -.05123 & -.05926 & .05029 & -.03643 & .05465 \\
-.3333 & .1303 & -15.69 & -.6362 & -.1585 & -.1025 & -.1185 & .1006 & -.07825 & .1093 \\
-.3333 & .1236 & -.1560 & -.3368 & -.01583 & -.1537 & -.1778 & .1509 & -.1093 & .1639 \\
-.3333 & .1169 & .1123 & -.03750 & .1269 & -.2049 & -.2371 & .2012 & -.1457 & .2186 \\
-.3333 & .1169 & .01779 & -.03750 & -.03936 & -.2562 & -.2963 & .2515 & -.1821 & .2732 \\
-.3333 & .1169 & .01779 & -.03750 & -.03936 & -.3074 & -.3556 & .3017 & -.2186 & .3279 \\
-.3333 & .1169 & .01779 & -.03750 & -.03936 & -.3586 & -.3903 & .3814 & .1583 & .3019 \\
-.3333 & .1169 & .01779 & -.03750 & -.03936 & -.4098 & -.4251 & .3761 & .5352 & .3919 \\
-.3333 & .1169 & .01779 & -.03750 & -.03936 & -.4611 & -.4251 & .4965 & .5352 & .4977 \\
-.3333 & .1169 & .01779 & -.03750 & -.03936 & -.5123 & -.4251 & .4965 & .5352 & .4977
\end{bmatrix} \tag{34}$$

$$\phi^{-1} = \begin{bmatrix}
-.3226 & -.03536 & 1.415 & -2.775 & -14.94 & 13.66 & 0 & 0 & 0 & 0 \\
-.9261 & -.0656 & 2.909 & -5.738 & 7.640 & -3.820 & 0 & 0 & 0 & 0 \\
1.938E-6 & -.06552 & .1310 & -.06551 & -1.599E-5 & 7.997E-6 & 0 & 0 & 0 & 0 \\
-.02070 & .03951 & -3.358 & .3856 & 9.248 & -6.294 & 0 & 0 & 0 & 0 \\
0 & .03727 & -.07454 & 6.053 & -12.03 & 6.015 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 19.52 & -19.52 \\
0 & 0 & 0 & 0 & 15.97 & -14.33 & -1.739 & -7.665 & -1.988 & 9.750 \\
0 & 0 & 0 & 0 & 3.917E-7 & -4.609 & 9.218 & -9.663 & 10.11 & -5.054 \\
0 & 0 & 0 & 0 & 1.472 & -2.638 & -.1788 & 1.442 & .8440 & -.9415 \\
0 & 0 & 0 & 0 & -1.183E-7 & 5.244 & -10.49 & 1.542 & 7.403 & -3.702
\end{bmatrix} \tag{35}$$

$$G_1 = \begin{bmatrix}
2.0 & -3.240 & 8.168 \\
.0 & 2.000 & 14.13 \\
.0 & -1.858E-10 & 2.0
\end{bmatrix} \tag{36}$$

$$G_2 = \begin{bmatrix} 2.0 & -.5554 \\ .0 & 2.0 \end{bmatrix} \tag{37}$$

$$G_3 = [1.0] \tag{38}$$

$$G_4 = \begin{bmatrix} 3.0 & -3.465 \\ .0 & 3.0 \end{bmatrix} \tag{39}$$

$$G_5 = \begin{bmatrix} 3.0 & .2807 \\ .0 & 3.0 \end{bmatrix} \tag{40}$$

The computed eigenvalues are

$$\begin{array}{ll} \lambda_1 = 2.000059 & \lambda_6 = 1.000000 \\ \lambda_2 = 1.999970 + i5.123743E\text{-}5 & \lambda_7 = 3.000000 \\ \lambda_3 = 1.999970 - i5.123743E\text{-}5 & \lambda_8 = 3.000000 \\ \lambda_4 = 2.000000 & \lambda_9 = 3.000000 \\ \lambda_5 = 2.000000 & \lambda_{10} = 3.000000 \end{array} \tag{41}$$

Note the nonexactness of λ_1, λ_2 and λ_3 is not a problem in BLOCK IT; these are the eigenvalues which are computed via standard EISPACK procedures. The matrix A is simply such that its eigenvalues are difficult to compute with extreme precision. The interesting point is that even though eigenvalues were not computed with desired accuracy, ϕ, ϕ^{-1} and all G_i are still computed to an accuracy defined by equation 33.

The reader is referred to Reference 5 for many more test cases. The program tape contains the results of applying BLOCK IT to 13 of the 27 test problems given in Reference 12, and several other tests.

Contained within the "other tests" in Reference 5 is the solution to a matrix which is set up in the form of equation 21, and in the form of equation 25. The eigenvalues are known a priori to be -1,-2,...,-22. The object of the test was to compare blocking achievable for matrices in Frobenius form versus other forms. As expected, BLOCK IT failed for equation 21. For equation 25, it was perfectly successful in that the 22 X 22 matrix was blocked out as 22 1 X 1 blocks with each column of ϕ being equal to the expected eigenvector defined via equation 26. The failure results for equation 21 were such that nearly all quasi-eigenvectors were nearly parallel. This was not surprising in light of equation 22.

CONCLUSION

An algorithm that can be used to obtain reduced order feedback control equations for both time and frequency domain analysis has been developed and discussed. In a more general context, the algorithm computes a nonsingular similarity transformation matrix with real coefficients, which can be used to reduce a nonsymmetric real matrix to block diagonal form. Each block is a real quasi-upper traingular matrix. The algorithm works with both defective and derogatory matrices. When and if it fails, the resultant output can be used as a guide for reformulation of the mathematical equations that lead up to the ill-conditioned matrix that could not be block diagonalized.

REFERENCES

1) Bartels, R.H. and Stewart, G.W., "Algorithm 432, The Solution of the Matrix Equation AX - XB = C," *Communications of the Association for Computing Machinery*, 1972, pp. 820-826.

2) Bavely, C.A. and Stewart, G.W., "An Algorithm for Computing Reducing Subspaces by Block Diagonalization," *SIAM Journal of Numerical Analysis*, Vol. 16, No. 2, April 1979.

3) Bodley, C.S., Devers, A.D., Park, A.C., and Frisch, H.P., "A Digital Computer Program for the Dynamic Interaction Simulation of Controls and Structure (DISCOS)," NASA Technical Paper 1219, Vols. 1 and 2; (available COSMIC, University of Georgia, Athens, Georgia). Program GSC-12422, May 1978.

4) DiStefano, JJ., III, Stubberud, A.R., and Williams, I.J., Theory and Problem of Feedback and Control Systems, *Schaum's Outline Series*, McGraw Hill, 1967.

5) Frisch, H.P., "BLOCK IT; an algorithm for obtaining reduced order dynamic models," GSFC Computer Program Library, Program Number N00160; also available COSMIC, University of Georgia, Athens, Georgia, GSC-12723.

6) Frisch, H.P., "Time and Frequency Domain Analysis of Sampled Data Controllers via Linear Mixed Operation Equations," NASA Technical Paper #1817, October 1980.

7) Frisch, H.P., "Pitfalls and Guidelines for the Numerical Evaluation of Moderate Order System Frequency Response," NASA Technical Paper #1814, October 1980.

8) Garbow, B.S., Boyle, J.M., Dongarra, J.J., and Moler, C.B., *Matrix Eigensystem Routines – EISPACK Guide Extension*, Vol. 51, Springer-Verlag, 1977.

9) Golub, G.H. and Wilkinson, J.H., "Ill-Conditioned Eigensystems and the Computation of the Jordan Canonical Form," *SIAM Review*, Vol. 18, No. 6, October 1976.

10) Graham, A., *Matrix Theory and Applications for Engineers and Mathematicians*, Ellis Harwood Limited, division of John Wiley, 1979.

11) Gregory, R.T., "Defective and Derogatory Matrices," *SIAM Review*, Vol. 2, No. 2, April 1960, pp. 134-139.

12) Gregory, R.T. and Karney, D.L., *A Collection of Matrices for Testing Computational Algorithms*, Wiley-Interscience, 1969.

13) Hamming, R.W., *Numerical Methods for Scientists and Engineers*, McGraw Hill, 1973.

14) Klema, V.C. and Laub, A.J., "The Singular Value Decomposition: Its Computation and Some Applications," *IEEE Transformation on Automatic Control*, Vol. AC-25, No. 2, April 1980, pp. 164-176.

15) Kuo, B.C., *Automatic Control Systems*, Prentice Hall, Inc., 1967.

16) MacFarlane, A.G.J., "Frequency-Response Methods in Control Systems," *IEEE Press*, 1979.

17) Moler, C. and Van Loan, C., "Nineteen Dubious Ways to Compute the Exponential of a Matrix," *SIAM Review*, Vol. 20, No. 4, October 1978.

18) Pease, M.C., *Methods of Matrix Algebra*, Academic Press, 1965.

19) Smith, B.T., Boyle, J.M., Dongarra, J.J., Garbow, B.S., Ikebo, Y., Klema, V.C., and Moler, C.B., *Matrix Eigensystem Routines – EISPACK Guide*, Vol. 6, Springer-Verlag, 1976.

20) Stewart, G.W., "Algorithm 506 HQR3 and EXCHNG: Fortran Subroutines for Calculating and Ordering the Eigenvalues of a Real Upper Hessenberg Matrix," *ACM Transformation on Mathematical Software*, Vol. 2, No. 3, September 1976, pp. 275-280.

21) Wilkinson, J.H. and Rensch, C., *Handbook for Automatic Computation, Linear Algebra,* Vol. II, Springer-Verlag, 1971.

22) Wilkinson, J.H., *The Algebraic Eigenvalue Problem*, Oxford Press, 1965.

1. Report No. NASA TP-1818	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Reduced Order Feedback Control Equations for Linear Time and Frequency Domain Analysis		5. Report Date June 1981 6. Performing Organization Code 712
7. Author(s) Harold P. Frisch		8. Performing Organization Report No.
9. Performing Organization Name and Address Goddard Space Flight Center Greenbelt, Maryland 20771		10. Work Unit No. 11. Contract or Grant No. 13. Type of Report and Period Covered Technical Paper
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

An algorithm has been developed and discussed which can be used to obtain reduced order feedback control equations for both time and frequency domain analysis. In a more general context, the algorithm computes a real nonsingular similarity transformation matrix which reduces a real nonsymmetric matrix to block diagonal form, each block of which is a real quasi-upper triangular matrix. The algorithm works with both defective and derogatory matrices and when and if it fails, the resultant output can be used as a guide for the reformulation of the mathematical equations that lead up to the ill-conditioned matrix which could not be block diagonalized.

17. Key Words (Selected by Author(s))	18. Distribution Statement
Analytical and Numerical Methods; Guidance and Control; Spacecraft Simulation	Unclassified - Unlimited Subject Category 18

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	27	A03

*For sale by the National Technical Information Service, Springfield, Virginia 22161.

NASA-Langley, 1981