NASA TECHNICAL NOTE

NASA TN D-6860

N72-27002

NASA TN D-6860

AN EFFICIENT ALGORITHM USING MATRIX METHODS TO SOLVE WIND-TUNNEL FORCE-BALANCE EQUATIONS

by David L. Smith

Langley Research Center
Hampton, Va. 23365

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • AUGUST 1972

1. Report No. NASA TN D-6860	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle AN EFFICIENT ALGORITHM USING MATRIX METHODS TO SOLVE WIND-TUNNEL FORCE-BALANCE EQUATIONS		5. Report Date August 1972 6. Performing Organization Code
7. Author(s) David L. Smith		8. Performing Organization Report No. L-8278
9. Performing Organization Name and Address NASA Langley Research Center Hampton, Va. 23365		10. Work Unit No. 501-06-10-05 11. Contract or Grant No.
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546		13. Type of Report and Period Covered Technical Note 14. Sponsoring Agency Code

15. Supplementary Notes

The information presented herein is largely based on a thesis entitled "The Application of Matrix Methods to Solving Wind-Tunnel Force-Balance Equations" submitted by the author to the Faculty of the School of Engineering and Applied Science of George Washington University in partial satisfaction of the requirements for the degree of Master of Science, December 1971.

16. Abstract

An iterative procedure applying matrix methods to accomplish an efficient algorithm for automatic computer reduction of wind-tunnel force-balance data has been developed. Balance equations are expressed in a matrix form that is convenient for storing balance sensitivities and interaction coefficient values for online or offline batch data reduction. The convergence of the iterative values to a unique solution of this system of equations is investigated, and it is shown that for balances which satisfy the criteria discussed, this type of solution does occur. Methods for making sensitivity adjustments and initial load effect considerations in wind-tunnel applications are also discussed, and the logic for determining the convergence accuracy limits for the iterative solution is given.

This more efficient data reduction program is compared with the technique presently in use at the NASA Langley Research Center, and computational times on the order of one-third or less are demonstrated by use of this new program.

17. Key Words (Suggested by Author(s)) Force-balance data reduction Matrix methods Wind-tunnel instrumentation	18. Distribution Statement Unclassified – Unlimited		
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 43	22. Price* $3.00

*For sale by the National Technical Information Service, Springfield, Virginia 22151

AN EFFICIENT ALGORITHM USING MATRIX METHODS TO SOLVE WIND-TUNNEL FORCE-BALANCE EQUATIONS*

By David L. Smith
Langley Research Center

SUMMARY

An iterative procedure applying matrix methods to accomplish an efficient algorithm for automatic computer reduction of wind-tunnel force-balance data has been developed. Balance equations are expressed in a matrix form that is convenient for storing balance sensitivities and interaction coefficient values for online or offline batch data reduction. The convergence of the iterative values to a unique solution of this system of equations is investigated, and it is shown that for balances which satisfy the criteria discussed, this type of solution does occur. Methods for making sensitivity adjustments and initial load effect considerations in wind-tunnel applications are also discussed, and the logic for determining the convergence accuracy limits for the iterative solution is given.

This more efficient data reduction program is compared with the technique presently in use at the NASA Langley Research Center, and computational times on the order of one-third or less are demonstrated by use of this new program.

INTRODUCTION

Since aircraft and space vehicle motions depend on the forces and moments about the three orthogonal body axes, an extensive amount of wind-tunnel testing is devoted to measuring these quantities for given model configurations to enable the estimation of aerodynamic loads on full-scale vehicles in flight. The most commonly used method for measuring these forces and moments is by installing an internal strain-gage balance within a wind-tunnel model as illustrated in figure 1. The model is attached to this balance and the forces and moments about the axes shown in figure 2 are transduced into electrical signals suitable for analog-to-digital conversion and subsequent data reduction or online evaluation where such equipment exists.

---

*The information presented herein is largely based on a thesis entitled "The Application of Matrix Methods to Solving Wind-Tunnel Force-Balance Equations" submitted by the author to the Faculty of the School of Engineering and Applied Science of George Washington University in partial satisfaction of the requirements for the degree of Master of Science, December 1971.

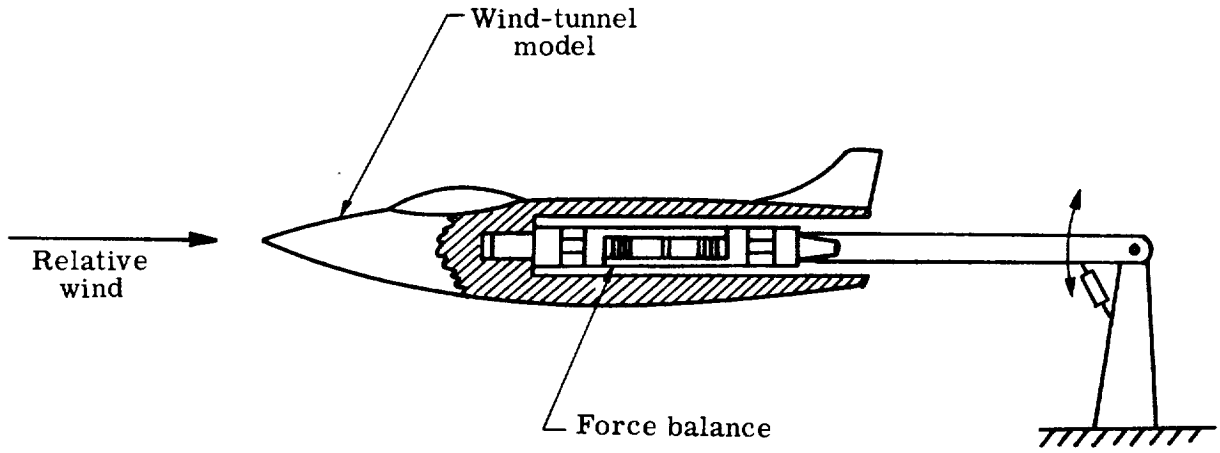


Figure 1.- Typical installation of internal strain-gage balance in a wind-tunnel model.

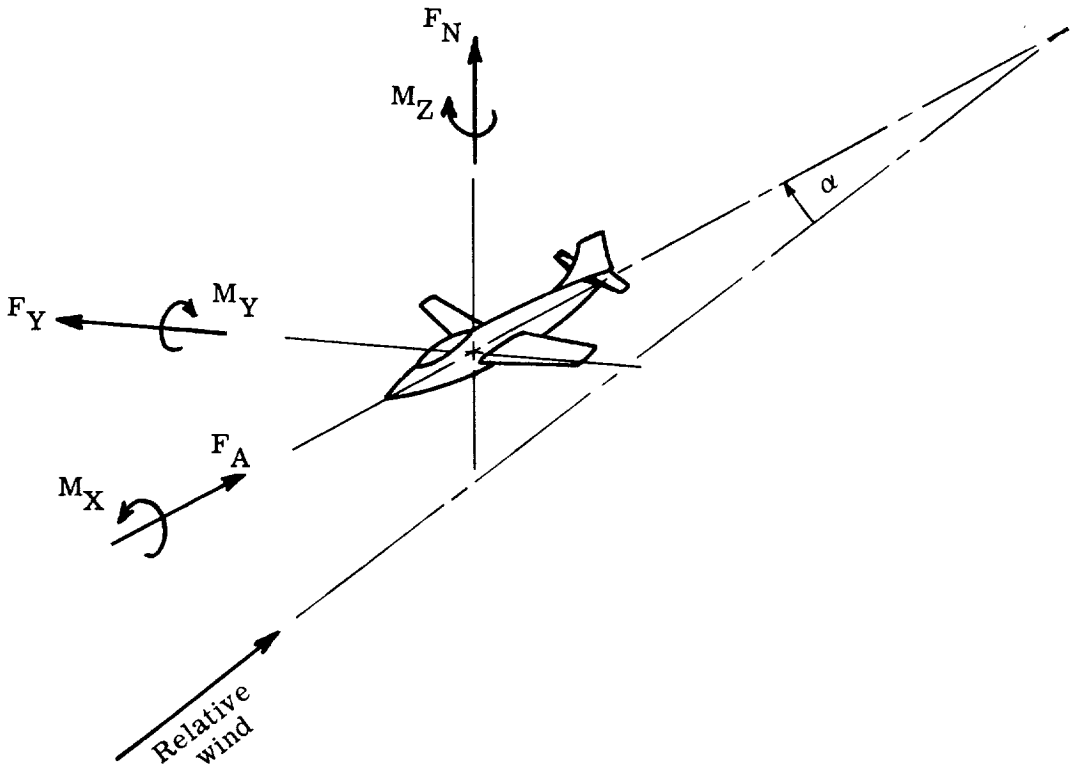


Figure 2.- Force and moment axes with positive directions shown.

The purpose of this investigation is to apply matrix methods to the force-balance equations in order to develop an efficient data reduction program which offers significantly fewer arithmetic operations and smaller computational times per data point. This program uses an iterative procedure to account for balance interactions and considers required sensitivity adjustments and initial load effects. A description of the balance data reduction is given and a technique is presented for determining from calibration data whether the iterative procedure will converge. The technique presently in use at the Langley Research Center is presented in appendix A.

SYMBOLS

Measurements are given in both SI and U.S. Customary Units. The measurements and calculations were made in U.S. Customary Units. The force and moment axes usually coincide with the body axes of a wind-tunnel model as shown in figure 2.

c	element of matrix M
F_A	axial force
F_N	normal force
F_Y	side force
f	generalized force or moment function
J	upper bound for Lipschitz' constant
K	normalized coefficient
k	calibration coefficient
M_X	rolling moment
M_Y	pitching moment
M_Z	yawing moment
X	generalized force or moment component
α	angle of attack

ϵ	nonlinear interaction correction
θ	meter indication
κ	sensitivity

Matrices:

B	positive full-scale balance design loads matrix (6×1)
C_1	first-order coefficient matrix (6×6)
C_2	nonlinear interaction coefficient matrix (6×21)
E	second-order interaction column matrix (6×1)
F	force and moment column matrix (6×1)
$\mathcal{K}$	sensitivity diagonal matrix (6×6)
M	matrix product of $C_1^{-1}C_2$, (6×21)
P	force and moment product matrix (21×1)
Θ	output column matrix (6×1)

Subscripts:

i	force or moment component considered
j	interacting load (see table I)
k	data point specified
max	maximum value
min	minimum value
n	number of iteration

o initial load effect

u uncorrected value

DEVELOPMENT OF MATRIX RELATIONS

Background for Analysis

Ideally, the output for each force or moment component measured by a balance should be affected only by a loading on that particular component. Experience shows, however, that a given component is often affected by loading another component. This effect is called an "interaction." Interactions are classified as either linear or nonlinear, depending on whether they are related to a single component's load or to exponential powers and combinations of the components being loaded. Linear interactions result from machining tolerances, strain-gage positioning tolerances, variations in strain-gage properties, electrical circuitry, or Poisson's effect. Nonlinear interactions are attributed to deflections of the strain-gage beams (ref. 1).

Consider the general case of a balance designed to measure three perpendicular forces and three moments about the axes of these forces. The output of each component is a function of all six forces and moments due to presence of interactions and can be expressed as a polynomial of the form (ref. 2):

$$\theta_i = k_{i,1}F_N + k_{i,2}F_A + k_{i,3}M_Y + \ldots + k_{i,6}F_Y + k_{i,7}F_N^2 + k_{i,8}F_NF_A + k_{i,9}F_NM_Y + \ldots + k_{i,27}F_Y^2 + k_{i,28}F_N^3 + k_{i,29}F_N^2F_A + \ldots \quad (1)$$

The linear interaction coefficients for this case would be $k_{i,1}, k_{i,2}, \ldots, k_{i,6}$ or the coefficients of the first-order terms of equation (1), except for the $k_{i,i}$ which is the inverse of the ith component's sensitivity. Nonlinear interaction coefficients would be $k_{i,7}, k_{i,8}, \ldots$, or coefficients of second degree and higher order terms. In practice, third and higher order interaction terms are negligible, and second-order terms are generally small compared to the linear terms. (See refs. 1 and 2.)

To facilitate force-balance data reduction, equation (1) is divided through by $k_{i,i}$ or "normalized," with third and higher order terms neglected, yielding (ref. 2):

$$(X_i)_u = K_{i,1}F_N + K_{i,2}F_A + K_{i,3}M_Y + \ldots + K_{i,6}F_Y + K_{i,7}F_N^2 + K_{i,8}F_NF_A + K_{i,9}F_NM_Y + \ldots + K_{i,27}F_Y^2 \quad (2a)$$

where

$$K_{i,j} = \frac{k_{i,j}}{k_{i,i}} = \text{Normalized interaction coefficients when } i \neq j$$

with

$$\kappa_{i,i} = \frac{1}{k_{i,i}} = \text{ith component sensitivity when } i = j$$

and

$$\left(X_i\right)_u = \kappa_{i,i}\theta_i = \text{Uncorrected force or moment on ith component}$$

For a typical balance load, for example, normal force, $\left(X_i\right)_u = \left(F_N\right)_u$ and $K_{1,1} = \frac{k_{1,1}}{k_{1,1}} = 1$ which results in the following form of equation (2):

$$F_N = \left(F_N\right)_u - \left(K_{1,2}F_A + K_{1,3}M_Y + \ldots + K_{1,6}F_Y + K_{1,7}F_N^2 + K_{1,8}F_NF_A + K_{1,9}F_NM_Y + \ldots + K_{1,27}F_Y^2\right) \tag{2b}$$

In this form the interaction coefficients are expressed in terms of the apparent load on the ith component per unit of the jth loading.

Assumptions

In order to solve the system of six force-balance equations represented by equations (2) for the aerodynamic loads acting on a wind-tunnel model, the interaction coefficients and sensitivity constants used in acquiring the data must be known. It will be assumed that these constants are available from the calibration of the balance, and that the sensitivities have been adjusted to the actual values in the tunnel installation. It will also be assumed that there are no initial load effects to account for at this time. Methods for including both the balance sensitivity adjustments and the initial load effects will be considered under a subsequent heading.

Matrix Relations

By defining the following matrices, equation (2a) can be expressed as a matrix relation where the subscripts denote the loads indicated in table I:

TABLE I.- LOAD CORRESPONDING TO A GIVEN SUBSCRIPT

Load	Load denoted by subscript –	Load	Load denoted by subscript –
F_N	1	$F_A M_X$	15
F_A	2	$F_A M_Z$	16
M_Y	3	$F_A F_Y$	17
M_X	4	M_Y^2	18
M_Z	5	$M_Y M_X$	19
F_Y	6	$M_Y M_Z$	20
F_N^2	7	$M_Y F_Y$	21
$F_N F_A$	8	M_X^2	22
$F_N M_Y$	9	$M_X M_Z$	23
$F_N M_X$	10	$M_X F_Y$	24
$F_N M_Z$	11	M_Z^2	25
$F_N F_Y$	12	$M_Z F_Y$	26
F_A^2	13	F_Y^2	27
$F_A M_Y$	14		

The output column matrix:

$$\Theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \cdot \\ \cdot \\ \cdot \\ \theta_6 \end{bmatrix} = \left[\theta_i\right] \qquad (i = 1, 2, \ldots, 6)$$

The force and moment column matrix:

$$F = \begin{bmatrix} F_N \\ F_A \\ M_Y \\ M_X \\ M_Z \\ F_Y \end{bmatrix} = \begin{bmatrix} X_i \end{bmatrix} \qquad (i = 1, 2, \ldots, 6)$$

The second-order force and moment product column matrix:

$$P = \begin{bmatrix} F_N^2 \\ F_N F_A \\ F_N M_Y \\ \cdot \\ \cdot \\ \cdot \\ F_Y^2 \end{bmatrix} = \begin{bmatrix} X_i X_j \end{bmatrix} \qquad \begin{pmatrix} i = 1, 2, \ldots, 6 \\ j = i, i+1, \ldots, 6 \end{pmatrix}$$

The 6×6 sensitivity diagonal matrix:

$$\mathcal{K} = \begin{bmatrix} \kappa_{1,1} & 0 & 0 & 0 & 0 & 0 \\ 0 & \kappa_{2,2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \kappa_{3,3} & 0 & 0 & 0 \\ 0 & 0 & 0 & \kappa_{4,4} & 0 & 0 \\ 0 & 0 & 0 & 0 & \kappa_{5,5} & 0 \\ 0 & 0 & 0 & 0 & 0 & \kappa_{6,6} \end{bmatrix}$$

The 6×6 first-order interaction coefficient matrix:

$$C_1 = \begin{bmatrix} 1 & K_{1,2} & K_{1,3} & K_{1,4} & K_{1,5} & K_{1,6} \\ K_{2,1} & 1 & K_{2,3} & K_{2,4} & K_{2,5} & K_{2,6} \\ K_{3,1} & . & . & . & . & . \\ . & . & . & . & . & . \\ K_{6,1} & K_{6,2} & K_{6,3} & K_{6,4} & K_{6,5} & 1 \end{bmatrix}$$

The 6×21 second-order interaction coefficient matrix:

$$C_2 = \begin{bmatrix} K_{1,7} & K_{1,8} & K_{1,9} & \cdots & K_{1,27} \\ K_{2,7} & K_{2,8} & K_{2,9} & \cdots & . \\ . & . & . & \cdots & . \\ K_{6,7} & K_{6,8} & K_{6,9} & \cdots & K_{6,27} \end{bmatrix}$$

By use of these matrices, the six equations represented by equation (2) can be expressed as follows:

$$\chi\Theta = C_1F + C_2P \tag{3}$$

By use of matrix algebra (ref. 3), equation (3) is readily solved for F by subtracting C_2P from each side and premultiplying each term by the matrix inverse of C_1 to obtain

$$F = C_1^{-1}\chi\Theta - C_1^{-1}C_2P \tag{4}$$

In this form, equation (4) is very convenient for using iterative procedures to solve for F, or the forces and moments acting on the model. An iterative procedure is the most practical method of solving this equation because the second-order interaction corrections $C_1^{-1}C_2P$ are functions of the elements of F.

Since the previously defined coefficient matrices are made up of constants determined from calibration data, C_1^{-1} and the product of $C_1^{-1}C_2$ can be calculated and stored in this form for subsequent data reduction. Also, if a balance is designed to measure less than six components, the coefficient matrices can be accordingly reduced in size before these calculations are made. Carrying out these steps prior to actually reducing tunnel data greatly increases the efficiency of the data reduction program.

Iterative Procedures

Upon examination, equation (4) tends to appear cumbersome or awkward to solve iteratively as each term on the right-hand side is a product of three matrices. Further examination shows this is not the case; actually, it is in a rather convenient form for the data reduction program. The product $\mathcal{K}\Theta$, or the engineering unit conversion, is calculated prior to the iteration stage of the data reduction and is called the "uncorrected" load, F_u. Also, the product of $C_1^{-1}C_2$ is calculated from the calibration data and is stored as a 6×21 matrix, M. Consequently, equation (6) can be expressed as follows:

$$F = C_1^{-1}F_u - MP \tag{5}$$

For a given data point Θ_k, $F_1 = C_1^{-1}F_u$ is directly calculable and is a very good approximation of F since F_1 contains all first-order interaction corrections and because of the relatively small effects of second-order balance interactions. For this reason, the elements of F_1 are used as the values of the forces and moments necessary for calculating the elements of the first approximation of the second-order matrix. Note that $F_1 = C_1^{-1}K_1\Theta$ is dependent only on calibration constants and on the meter indications for the particular data point being reduced. These linear terms are directly calculated and require no iterating for their evaluation. Only the second-order interaction terms are iterated until the procedure converges.

Iteration of the second-order terms is accomplished as follows: The first approximation of the second-order matrix P_1 is premultiplied by M and generates the second-order interaction correction matrix E_1. The absolute value of each E_1 element $|\epsilon_i|$ is then compared with the required accuracy of convergence for each balance component. If each $|\epsilon_i|$ is less than the given convergence limit, which can be specified or is calculated based on the balance sensitivities, the data reduction is complete for that data point with $F = F_1 - MP_1$. However, if one or more of the $|\epsilon_i|$ is greater than these convergence accuracies, equation (5) is reiterated as follows:

$$F_n = C_1^{-1}F_u - E_{n-1} \tag{6}$$

The quantity F_n is then used to reevaluate P_n which is again premultiplied by M to determine more nearly exact values of the second-order balance interactions E_n. The column matrix E_n is then compared with E_{n-1} and if corresponding elements agree to within the convergence limits, the force and moment matrix may be expressed as

$$F = C_1^{-1}F_u - E_n \tag{7}$$

Equation (6) is reiterated until convergence occurs.

Convergence

In any iterative solution such as the one described, certain questions must be considered such as whether F_n will always converge, and whether its limit is a unique solution of the given equations. Henrici (ref. 4) considers these questions for the general case of iterating systems of nonlinear equations. A criterion for proving that equation (5) will converge is given and discussed in appendix B. Theorems given therein not only prove that this iterative procedure will converge but also show that successive iterations approach a unique solution in the region of the design loads for balances which satisfy the given conditions and have an upper bound for Lipschitz' constant J of less than 1 where

$$J = \max_{F \in R} \sqrt{\left(\frac{\partial f_1}{\partial F_N}\right)^2 + \left(\frac{\partial f_1}{\partial F_A}\right)^2 + \ldots + \left(\frac{\partial f_i}{\partial X_j}\right)^2 + \ldots + \left(\frac{\partial f_6}{\partial F_Y}\right)^2} \qquad \begin{pmatrix} i = 1, 2, \ldots, 6 \\ j = 1, 2, \ldots, 6 \end{pmatrix} \tag{8}$$

A computer subroutine program which calculates the value of J is given in appendix B.

CONSIDERATIONS FOR REDUCING BALANCE DATA

Sensitivity Adjustments

The reduction of force-balance data requires each force and moment component calibration sensitivity to be adjusted to the actual values for the wind-tunnel installation. To accomplish this sensitivity adjustment, the same apparent loading is applied to each component at the calibration facility and at the wind-tunnel installation. The ratio of the output at calibration to the output at the tunnel installation for this common apparent load is then multiplied by the corresponding calibration sensitivity and yields the tunnel sensitivity as follows:

$$\left(\kappa_{i,i}\right)_{\text{tunnel}} = \left[\left(\kappa_{i,i}\right)_{\text{calibration}}\right]\left(\frac{\theta_{\text{calibration}}}{\theta_{\text{tunnel}}}\right) \tag{9}$$

An efficient method for making this sensitivity adjustment is to store the calibration sensitivities and apparent load outputs with the balance interaction coefficient values. Then, adjusted tunnel sensitivities can readily be computed and assigned to the proper locations in the $\mathcal{K}$ matrix by supplying balance output values for the same apparent load in the tunnel installation, and performing the indicated ratios in the data reduction program.

Initial Loads

Another important consideration in force-balance data reduction stems from the fact that the balance equations (2) are nonlinear. For this reason, tunnel data must be related to the same origin as calibration data, or zero output for zero loads on all components, as shown by the solid line in figure 3. Typically, meter readings at wind-off zero-angle-of-attack conditions are taken as the zero load values or as the origin of the data. However, initial loads such as model weight cause the balance output to be located off the calibration origin, for example, at point A in figure 3 where the prime indicates the tunnel axis system. Ignoring initial load effects is essentially the same as assuming that the balance is performing according to the dashed curve or the calibration curve shifted to the new origin. The balance is actually performing according to the solid calibration curve. Therefore, the data origin must be shifted to correspond with the calibration origin for each data point.

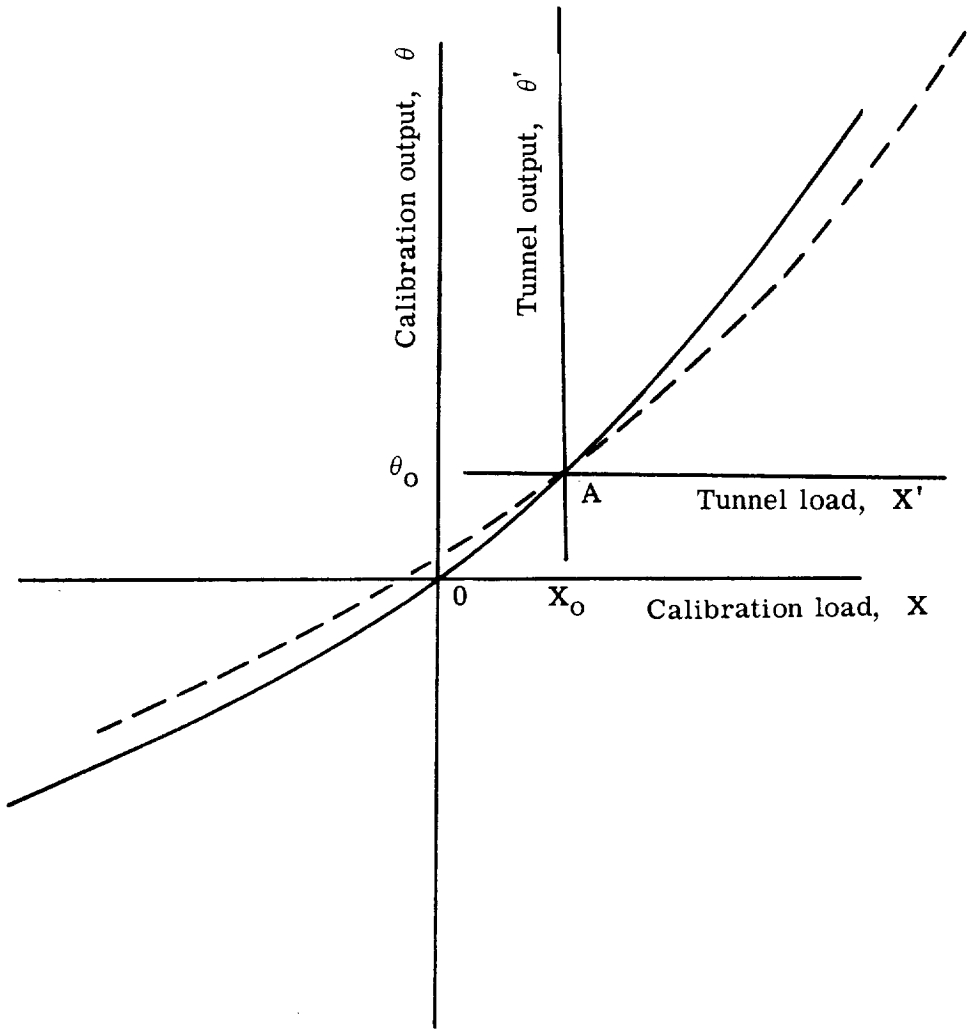


Figure 3.- Initial load effect on balance output.

A convenient method for reducing balance data with initial loads is to translate the axes to the system used for calibration, eliminate balance interaction effects, and then translate the axes back to the set used in acquiring the data. Note in figure 3, that θ_O and X_O are determined prior to taking aerodynamic data, and θ' is recorded for each data point during a wind-tunnel test. This observation suggests the following axes translation:

$$\theta = \theta' + \theta_O \tag{10}$$

Substituting this value into the balance equation allows the calculation of X, from which X' is readily determined by the following translation back to the primed axes:

$$X' = X - X_O \tag{11}$$

This method is readily extended for a six-component balance as shown in the following matrix relations:

$$\Theta = \Theta' + \Theta_O \tag{12}$$

This output matrix is then substituted into equation (5) and is solved iteratively as described for F, from which

$$F' = F - F_O \tag{13}$$

This method of translating axes to include initial load corrections in balance data reduction has been used with the iterative procedure discussed previously. The only arithmetic operations required for these axes transfers are six additions before iterating the balance equations, and six subtractions following the iterations.

An alternate method for considering initial loads by reevaluating balance interaction coefficients to account for these axes translations is described in reference 5.

Convergence Limits

For any iterative solution, an accuracy or convergence limit must be specified. This limit can be an absolute value, as presently used with balance data at Langley, a percentage of the solution itself, a percentage of the maximum range of the solution, or a percentage of the resolution of the data acquisition system. Because of the limitation of the recording system resolution, the minimum detectable increment of each component is equal to its sensitivity times 1 count, or

$$(\Delta X_i)_{min} = \kappa_{i,i} \times 1 = \kappa_{i,i} \tag{14}$$

Consequently, the convergence limit of one-tenth the value of this minimum detectable increment, or $\kappa_{i,i}/10$, for each force and moment component is used in this data reduction program. This criterion will cause to be negligible any systematic errors that may result because of the convergence accuracy.

Data Reduction Program

The iterative procedure and other related topics that must be considered for balance data reduction have been utilized in developing a FORTRAN program for the Control Data 6600 digital computer complex at the Langley Research Center. This program is listed and described in appendix C.

COMPARISON OF PRESENT DATA REDUCTION METHOD WITH MATRIX METHODS

Logic

The logic of the matrix method developed in this paper and of the technique presently in use at the Langley Research Center (appendix A) is very similar in many ways. Both methods apply an iterative solution of the form

$$F_n = \underline{f}(F_{n-1}) \tag{15}$$

where $\underline{f}$ denotes a column vector. (See appendix B.) The presently used method iterates each force and moment component individually and updates or recalculates the second-order products between each component's iteration. The matrix method, however, iterates the column matrix F_n or updates all force and moment components before the second-order product matrix is recalculated. Also, the first approximation of these two methods is determined differently. The present method uses the products of the sensitivities times the balance outputs or $\kappa_{i,i}\theta_i$ and iterates both first- and second-order interaction terms, but the matrix method uses $C_1^{-1}\chi\theta$ as the first approximation and consequently must iterate only the second-order terms.

Computation Time Requirements

Because of the differences described, the number of arithmetic operations required per data point by using the matrix method is considerably reduced and, as a result, corresponding decreases in the computation time for each data point are obtained. Table II gives a comparison of the deviation of the first approximation iterated in the balance

equations for the two methods. These values are significantly closer to the calculated solutions for each balance component in the matrix method. Consequently, fewer iterations are required for the data points in using the matrix method as shown in three of the four cases presented in tables III and IV. There are also significant differences in the number of arithmetic operations for the two methods due not only to the few iterations but also to the facts that load combinations are updated after iterating all components and only second-order terms must be iterated by using the matrix method. This reduction in the number of arithmetic operations results in computation times on the order of one-third or less for the matrix method over the present technique. Table V shows considerable reductions in the number of arithmetic operations and computation times even when both methods are driven through the same number of iterations. These observations are especially noticeable for balances with less than six components because of the "collapsing" of the coefficient and data matrices as discussed previously. The matrix method is thereby significantly more efficient than the present technique in which the coefficients for components not measured are set equal to zero and the arithmetic operations are performed for all components and, as a result, there is the same number of calculations per iteration for any number of measured force and moment components.

TABLE II.- DEVIATION OF FIRST APPROXIMATION FROM SOLUTION FOR BOTH METHODS OF DATA REDUCTION

Balance component	Converged iterative solution	First approximation values		Deviation of approximation from iterative solution	
		Present method	Matrix method	Present method	Matrix method
F_N, N (lb)	350.5 (78.8)	343.0 (77.1)	350.1 (78.7)	7.5 (1.7)	0.4 (0.1)
F_A, N (lb)	53.8 (12.1)	67.6 (15.2)	55.6 (12.5)	13.8 (3.1)	1.8 (0.4)
M_Y, N-m (in-lb) . .	10.1 (89.6)	9.3 (82.0)	10.0 (88.9)	0.8 (7.6)	0.1 (0.7)
M_X, N-m (in-lb) . .	3.03 (26.8)	3.17 (28.1)	3.03 (26.8)	0.14 (1.3)	0 (0)
M_Z, N-m (in-lb) . .	5.65 (50.0)	5.62 (49.7)	5.74 (50.8)	0.03 (0.3)	0.09 (0.8)
F_Y, N (lb)	130 (29.2)	166 (37.3)	131 (29.4)	36 (8.1)	1 (0.2)

TABLE III.- COMPARISON OF MATRIX METHOD WITH PRESENT TECHNIQUE FOR TYPICAL DATA WITH NO INITIAL LOAD TRANSLATIONS

Number of balance components reduced	Number of iterations		Approximate number of arithmetic operations		Time required for iterations, msec	
	Present method	Matrix method	Present method	Matrix method	Present method	Matrix method
3	6	4	1728	105	16	2
4	3	3	864	166	12	2
5	3	2	864	205	10	4
6	5	2	1440	330	18	6

TABLE IV.- COMPARISON OF MATRIX METHOD WITH PRESENT TECHNIQUE FOR TYPICAL DATA WITH INITIAL LOAD TRANSLATIONS REQUIRED

Number of balance components reduced	Number of iterations		Approximate number of arithmetic operations		Time required for iterations, msec	
	Present method	Matrix method	Present method	Matrix method	Present method	Matrix method
3	6	4	1740	111	16	4
4	3	3	876	174	12	4
5	3	2	876	215	12	4
6	5	2	1452	342	18	4

TABLE V.- COMPARISON OF MATRIX METHOD WITH PRESENT TECHNIQUE FOR TYPICAL DATA POINTS WITH THE SAME NUMBER OF ITERATIONS FOR BOTH METHODS

Number of balance components reduced	Number of iterations		Approximate number of arithmetic operations		Time required for iterations, msec	
	Present method	Matrix method	Present method	Matrix method	Present method	Matrix method
3	7	7	2016	171	18	4
4	4	4	1152	208	12	4
5	3	3	864	295	10	6
6	4	4	1152	612	14	8

CONCLUDING REMARKS

The reduction of wind-tunnel force-balance data by applying matrix methods to an iterative solution of the balance equations has been presented, and it has been demonstrated that this method is a significant improvement over the presently used method at Langley Research Center. The convergence of this iterative solution was considered, and it was shown that for balance equations which satisfy the conditions specified, this method would converge to a unique solution within the range of the design loads of the balance. A technique was also presented to determine whether the balance equations satisfy these conditions based on the calibration data for the balance. Considerations of sensitivity adjustments and initial load effects were discussed and methods for making these corrections were given.

This matrix method has been developed with the assumption that the third and higher order balance interactions are negligible. If the case arises in which such interactions must be considered, this calculation can be readily accomplished with these methods by adding the load combination(s) which produce the third or higher order interaction to the force and moment product column matrix and including the appropriate coefficients on each row of the nonlinear interaction coefficient matrix. These changes would, of course, necessitate changing the dimensions of these arrays in the computer programs given herein.

The efficiency of this force-balance data reduction algorithm resulting from applying matrix methods makes it particularly useful for real-time display and control calculations by smaller online computers as well as beneficial for offline batch data reduction subsequent to the wind-tunnel test runs. Computational times of one-third or less than those required for the presently used technique are demonstrated by this matrix methods algorithm.

Langley Research Center,
National Aeronautics and Space Administration,
Hampton, Va., June 7, 1972.

APPENDIX A

A PRESENT METHOD OF FORCE-BALANCE DATA REDUCTION

The present method used to reduce force-balance data at the Langley Research Center involves an iterative solution of the six balance equations represented as follows:

$$\left(X_i\right)_u = K_{i,1}F_N + K_{i,2}F_A + K_{i,3}M_Y + \ldots + K_{i,6}F_Y + K_{i,7}F_N^2 + K_{i,8}F_NF_A + K_{i,9}F_NM_Y + \ldots + K_{i,27}F_Y^2 \quad \text{(A1)}$$

To solve these equations, the calibration sensitivity and all first- and second-order interaction coefficients must be known for each force and moment component.

After each force and moment component calibration sensitivity is adjusted to its actual value for the wind-tunnel installation, data reduction is accomplished through use of a computer subroutine program based on iterating equation (A1). The first approximation of each force and moment is obtained from

$$\left(X_i\right)_u = \kappa'_{i,i}\theta_i \quad \text{(A2)}$$

where the prime indicates that the tunnel sensitivity adjustment has been made. These first approximations of the aerodynamic loads are added to the initial load values and then substituted into equation (A1) to calculate a first approximate value of the interaction correction for each force and moment component. These correction values and the initial loads are then subtracted from the approximations of equation (A2), and the results become the second approximations of the aerodynamic loads. These second approximations are then added to the initial loads and reiterated into equation (A1), from which a second approximation to the interaction corrections is determined. The first and second interaction correction approximations are compared for each balance component and if they agree with a specified tolerance, then the latter corrections are subtracted from the force and moment approximations and the balance data reduction is complete. If these two approximations do not agree within the given tolerance, then the latter interaction corrections are subtracted from the force and moment approximations from equation (A2) and these values are reiterated into equation (A1) until convergence occurs for all balance components.

APPENDIX B

CONVERGENCE OF ITERATIVE SOLUTIONS

In order to develop a criterion for the convergence of the system of nonlinear balance equations, it will be convenient to use vector notation. The coordinates of the point $(F_N, F_A, M_Y, M_X, M_Z, F_Y)$ can be represented by the column vector F or

$$F = \begin{bmatrix} F_N \\ F_A \\ \cdot \\ \cdot \\ \cdot \\ F_Y \end{bmatrix} = \begin{bmatrix} f_1(F_N, F_A, M_Y, M_X, M_Z, F_Y) \\ f_2(F_N, F_A, M_Y, M_X, M_Z, F_Y) \\ \cdot \\ \cdot \\ \cdot \\ f_6(F_N, F_A, M_Y, M_X, M_Z, F_Y) \end{bmatrix} \tag{B1}$$

It is also convenient to denote a column vector with elements of $f_1, f_2, \ldots, f_6$ as $\underline{f}(F)$. Equation (B1) can thusly be written as follows:

$$F = \underline{f}(F) \tag{B2}$$

By using this notation, the following theorem given by Henrici (ref. 4) can be applied to the force-balance equations or to equation (5):

Theorem. Let R denote the region with limits a_i and b_i

$$a_1 \leqq F_N \leqq b_1$$

$$a_2 \leqq F_A \leqq b_2$$

$$a_3 \leqq M_Y \leqq b_3$$

$$a_4 \leqq M_X \leqq b_4$$

$$a_5 \leqq M_Z \leqq b_5$$

$$a_6 \leqq F_Y \leqq b_6$$

and let the functions $\underline{f}$ satisfy the following conditions:

(a) $f_1, f_2, \ldots, f_6$ are defined and continuous on R.

(b) For each $F \in R$, the point $f_1(F), f_2(F), \ldots, f_6(F)$ also lies in R.

(c) There exists a constant $L < 1$ such that for any two points F_1 and F_2 in R, the following inequality holds:

$$\left\| \underline{f}(F_1) - \underline{f}(F_2) \right\| \leq L \left\| F_1 - F_2 \right\| \tag{B3}$$

where the double bars denote the Euclidean norm. Then the following statements are true:

(a) Equation (B2) has precisely one solution S in R.

(b) For any choice of F_o in R, the limit of the iterative solution described or $F_n = \underline{f}(F_{n-1})$ is defined and converges to the unique solution S.

(c) For any $n = 1, 2, \ldots$, the following inequality holds:

$$\left\| F_n - S \right\| \leq \frac{L^n}{1 - L} \left\| F_1 - F_o \right\| \tag{B4}$$

It can easily be shown that the expressions of $f_1, f_2, \ldots, f_6$ satisfy conditions (a) and (b). Let the region R be bounded by 1.5 times the minimum and maximum loads for which a balance is designed to measure. The 1.5 factor is necessary as interaction effects can cause the first approximations to be outside of the design load region. Now consider f_1 expanded as follows:

$$f_1(F_N, F_A, M_Y, M_X, M_Z, F_Y) = (F_N)_1 + c_{1,7}(F_N)^2 + c_{1,8}F_N F_A + \ldots + c_{1,27}(F_Y)^2 \tag{B5}$$

where $(F_N)_1$ is the first approximation of normal force or the first element of $C_1^{-1}F_u$. For a given data point Θ_k, $(F_N)_1$ is constant. It is obvious that f_1 is continuous in R as are $f_2, \ldots, f_6$ (ref. 6), and by virtue of equations (B1) and (B5), condition (b) also is satisfied.

In order to establish that condition (c) is satisfied by the second-degree expressions $f_1, f_2, \ldots, f_6$, the Lipschitz constant L must be evaluated or a maximum for its value must be established. Henrici (ref. 4) has developed a criterion for determining the bound of the Lipschitz constant, which is given in the following theorem:

Let the functions $f_1, f_2, \ldots, f_6$ have continuous partial derivatives in the region R as defined. Then, the inequality (B3) holds with $L = J$, where

$$J = \max_{F \in R} \sqrt{\left(\frac{\partial f_1}{\partial F_N}\right)^2 + \left(\frac{\partial f_1}{\partial F_A}\right)^2 + \ldots + \left(\frac{\partial f_i}{\partial X_j}\right)^2 + \ldots + \left(\frac{\partial f_6}{\partial F_Y}\right)^2} \qquad \begin{pmatrix} i = 1, 2, \ldots, 6 \\ j = 1, 2, \ldots, 6 \end{pmatrix} \tag{B6}$$

The limiting value of J is calculated rather straightforwardly by taking the partial derivatives of the balance equations and evaluating the maximum possible value of each term as follows for $\frac{\partial f_1}{\partial F_N}$:

$$\frac{\partial f_1}{\partial F_N} = 2c_{1,7}F_N + c_{1,8}F_A + c_{1,9}M_Y + c_{1,10}M_X + c_{1,11}M_Z + c_{1,12}F_Y \tag{B7}$$

Each term on the right-hand side of equation (B7) is evaluated at 1.5 times the maximum design load, and the absolute values of these products are summed. This method of evaluation is carried out for each partial derivative in equation (B6) and results in an upper bound for J or

$$J \leq \sqrt{\left(\frac{\partial f_1}{\partial F_N}\right)^2_{max} + \left(\frac{\partial f_1}{\partial F_A}\right)^2_{max} + \ldots + \left(\frac{\partial f_i}{\partial X_j}\right)^2_{max} + \ldots + \left(\frac{\partial f_6}{\partial F_Y}\right)^2_{max}} \tag{B8}$$

A computer subroutine program which calculates the upper bound for J is listed. This program assumes that the $C_1^{-1}C_2$ product array is stored in M and that the design loads are stored in a one-dimension array B. The maximums of the partial derivatives are computed as indicated above and stored in the 6×6 array A from which the upper bound for J is calculated in accordance with equation (B8). The 1.5 factor is not applied in this subroutine but should be considered when interpreting the result of this evaluation. In practice this factor can be varied depending on the size of the interactions on the balance.

Because this upper limit of the Lipschitz constant is dependent only on the interaction coefficients and the design loads of a balance, it can be evaluated prior to the use of a balance in a wind-tunnel application. It is convenient to determine the bound of this constant at the same time the interaction coefficients are evaluated.

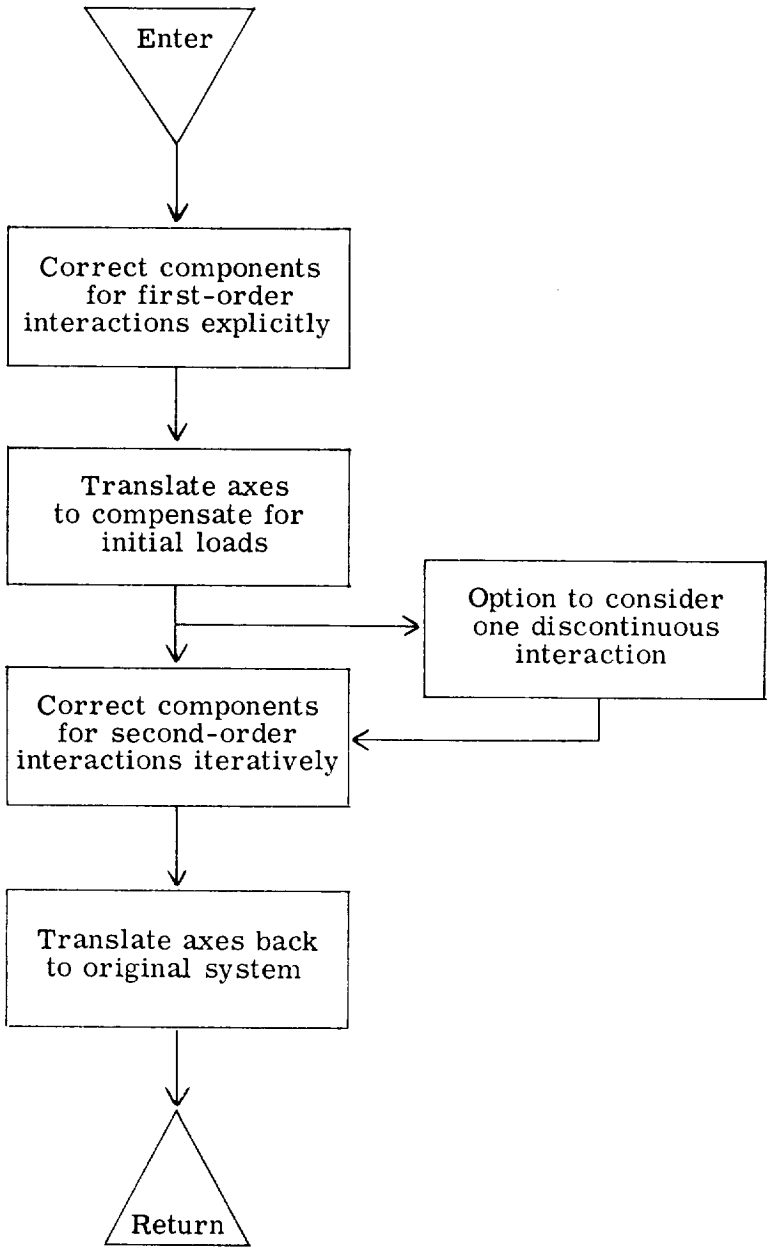
```
      SUBROUTINE LPSHZ(M,B,IBAL,IDATE)
      DIMENSION M(6,21) B(6)
C     THE VALUES OF REQUIRED PARTIAL DERIVATIVES WILL BE EVALUATED FROM
C     THE  DESIGN LOADS STORED IN B AND THE SECOND-ORDER INTERACTIONS
C     STORED IN M. THE PARTIAL DERIVATIVES WILL BE STORED IN A AND THE
C     LIPSCHITZ CONSTANT STORED IN ALIP.  IBAL AND IDATE ARE THE
C     BALANCE NAME AND CALIBRATION DATE RESPECTIVELY IN DISPLAY CODE.
      DO 10 I=1,6
      A(I,1)= (ABS(2.*M(I,1)*B(1))+ABS(M(I,2)*B(2))+ABS(M(I,3)*B(3))+
     1          ABS(   M(I,4)*B(4))+ABS(M(I,5)*B(5))+ABS(M(I,6)*B(6)))
      A(I,2)= (ABS(2.*M(I,7)*B(2))+ABS(M(I,2)*B(1))+ABS(M(I,8)*B(3))+
     1          ABS(   M(I,9)*B(4))+ABS(M(I,10)*B(1))+ABS(M(I,11)BS(6)))
      A(I,3)= (ABS(2.*M(I,12)*B(3))+ABS(M(I, 3)*B(1))+ABS(M(I, 8)*B(2))+
     1                              ABS(M(I,13)*B(4))+ABS(M(I,14)*B(5))+
     2                              ABS(M(I,15)*B(6)))
      A(I,4)= (ABS(2.*M(I,16)*B(4))+ABS(M(I, 4)*B(1))+ABS(M(I, 9)*B(2))+
     1                              ABS(M(I,13)*B(3))+ABS(M(I,17)*B(5))+
     2                              ABS(M(I,18)*B(6)))
      A(I,5)= (ABS(2.*M(I,19)*B(5))+ABS(M(I, 5)*B(1))+ABS(M(I,10)*B(2))+
     1                              ABS(M(I,14)*B(3))+ABS(M(I,17)*B(4))+
     2                              ABS(M(I,20)*B(6)))
      A(I,6)= (ABS(2.*M(I,21)*B(6))+ABS(M(I,6)*SB1))+ABS(BMI,11)*SB2))+
     1                              ABS(M(I,15)*B(3))+ABS(M(I,18)*B(4))+
     2                              ABS(M(I,20)*B(5)))
   10 CONTINUE
C     CALCUALTION OF LIPSCHITZ CONSTANT, ALIP
      ALIP=0.0
      DO 20 L=1,36
      ALIP=ALIP+A(L)**2
   20 CONTINUE
      ALIP=SQRT(ALIP)
      PRINT 15,IBAL,IDATE
15    FORMAT(1H1,///////* BALANCE   *,A10,*     DATE     *,A10)
      PRINT 16
16    FORMAT(1H0,*  THE FOLLOWING ARRAY CONTAINS THE PARTIAL DERIVATIVES
     1 OF X(J) WITH RESPECT TO X(I),*/* WHERE (I) DESIGNATES THE ROW AND
     2 (J) DESIGNATES THE COLUMN.*///)
      PRINT 17,A
17    FORMAT(1H0,6(2X,F10.6)/)
      PRINT 18,ALIP
18    FORMAT(1H0,* THE LIPSCHITZ CONSTANT FOR THIS BALANCE IS LESS THAN
     $*,F10.5)
      RETURN
      END
```

APPENDIX C

BALANCE DATA REDUCTION SUBROUTINE PROGRAMS

The subroutine programs given in this appendix correct force-balance data for interaction effects by applying the matrix methods discussed. Subroutine CTRNL calculates the initial load corrections necessary for determining second-order interactions on a multicomponent balance. Subroutine CINTR then corrects balance data for both first- and second-order interactions, considering initial load effects where required. Provisions are also included to account for one discontinuous interaction term, that is, an interaction coefficient for which the value depends on whether a particular component's load is positive or negative.

A flow chart of the subroutine CINTR follows. The listings of the two subroutine programs CTRNL and CINTR along with the required matrix operations subroutines are given with pertinent comments after the flow chart.


Enter
Correct components for first-order interactions explicitly
Translate axes to compensate for initial loads
Option to consider one discontinuous interaction
Correct components for second-order interactions iteratively
Translate axes back to original system
Return

```
      SUBROUTINE CINTR(FU,FZ,EZ,LIST,F,IER)
C
C     ****************************************************************
C     *                                                              *
C     *   SUBROUTINE                                                 *
C     *      CINTR                                                   *
C     *                                                              *
C     *   PURPOSE                                                    *
C     *      CORRECT MULTI-COMPONENT STRAIN GAGE BALANCE             *
C     *      RECORDINGS FOR 1ST AND 2ND ORDER INTERACTIONS           *
C     *                                                              *
C     *   ASSUMPTION                                                 *
C     *      THE BALANCE RECORDINGS HAVE BEEN CONVERTED              *
C     *      TO ENGINEERING UNITS. THAT IS, TUNNEL PRIME             *
C     *      SENSITIVITIES HAVE ALREADY BEEN APPLIED                 *
C     *                                                              *
C     *   LANGUAGE                                                   *
C     *      FORTRAN 2 OR 4                                          *
C     *                                                              *
C     *   USAGE                                                      *
C     *      DEFINE INPUT COMMON PARAMETERS                          *
C     *      CALL CINTR(FU,FZ,EZ,LIST,F,IER)                         *
C     *                                                              *
C     *   DESCRIPTION OF INPUT CALLING SEQUENCE PARAMETERS           *
C     *      FU      UNCORRECTED COMPONENTS, ENGINEERING UNITS       *
C     *      FZ      CORRECT INITIAL LOADS, DETERMINED ITERATIVELY   *
C     *      EZ      2ND ORDER INTERACTION DUE TO CORRECT INITIAL LOADS *
C     *      LIST    PRINT OPTION TO DISPLAY THE PATTERN OF CONVERGENCE *
C     *                   LIST=0 DO NOT PRINT COMPONENTS PER ITERATION *
C     *                   OTHERWISE, LIST IS THE LOGICAL UNIT NUMBER  *
C     *                                                              *
C     *   DESCRIPTION OF OUTPUT CALLING SEQUENCE PARAMETERS          *
C     *      F       COMPONENTS CORRECTED FOR INTERACTIONS           *
C     *      IER     ERROR INDICATOR FOR INTERACTION CONVERGENCE     *
C     *                   IER=0 INTERACTIONS CONVERGED               *
C     *                   IER=1 INTERACTIONS DID NOT CONVERGE        *
C     *                                                              *
C     *   DESCRIPTION OF INPUT COMMON PARAMETERS                     *
C     *      IBAL    BALANCE NAME IN DISPLAY CODE                    *
C     *              EACH BALANCE HAS BEEN ASSIGNED A UNIQUE NAME    *
C     *      IDATE   CALIBRATION DATE IN DISPLAY CODE                *
C     *              MONTH/DAY/YEAR  XX/YY/ZZ                        *
C     *      KDATE   CALIBRATION DATE EXPRESSED AS AN INTEGER        *
C     *              YEAR*10000+MONTH*100+DAY  ZZXXYY                *
C     *      M       NUMBER OF BALANCE COMPONENTS PHYSICALLY DEFINED *
C     *              M IS GREATER THAN 0, BUT LESS THAN OR EQUAL TO 6 *
C     *      NAMEC   ARRAY OF M COMPONENT NAMES IN A2 DISPLAY CODE   *
C     *              ALL MATRICES MUST BE ARRANGED ACCORDING TO NAMEC *
C     *      ITASK   INTEGER CODE SPECIFYING A TASK OR N1N2 TYPE BALANCE *
C     *              ALL MATRICES MUST BE CONSISTENT WITH THE CALIBRATION*
C     *      IORDR   ORDER OF THE BALANCE CALIBRATION                *
C     *              IORDR=0  NO INTERACTIONS                        *
C     *              IORDR=1  1ST ORDER INTERACTIONS ONLY            *
C     *              IORDR=2  1ST AND 2ND ORDER INTERACTIONS         *
C     *      ITRNL   OPTION TO TRANSLATE INTERACTIONS FOR INITIAL LOADS *
C     *              ITRNL=0  DO NOT TRANSLATE FOR INITIAL LOADS     *
C     *              ITRNL=1  DO TRANSLATE FOR INITIAL LOADS         *
C     *      IPLUS   OPTION FOR ONE 2ND ORDER DISCONTINUOUS INTERACTION *
C     *              IPLUS=0  NO DISCONTINUOUS INTERACTION TERM      *
C     *              OTHERWISE, NAMEC(IPLUS) IS THE ACTING COMPONENT *
C     *      MINUS   INDEX IN C1IC2 TO ACCOMMODATE DISCONTINUITY     *
C     *              NOTE,ONE 2ND ORDER TERM CHANGES ONE COLUMN OF C1IC2 *
C     *      NTRY    MAXIMUM NUMBER OF ITERATIONS ALLOWED FOR CONVERGENCE*
C     *              AN ERROR FLAG IS SET, IF NTRY IS INSUFFICIENT   *
```

```
C     *       C1I     INVERSE OF NORMALIZED 1ST ORDER INTERACTIONS WITH   *
C     *               MAIN DIAGONAL ELEMENTS OF 1. CONTAINS M*M ELEMENTS  *
C     *       C1IC2   PRODUCT OF C1I AND NORMALIZED 2ND ORDER INTERACTIONS*
C     *               CONTAINS M*N ELEMENTS, WHERE N=M(M+1)/2             *
C     *       CPOS    ARRAY OF M POSITIVE CALIBRATION CONSTANTS           *
C     *       CNEG    ARRAY OF M NEGATIVE CALIBRATION CONSTANTS           *
C     *       PRCNT   PERCENT ACCURACY REQUIRED FOR CONVERGENCE           *
C     *       CSENS   ARRAY OF M CALIBRATION PRIME SENSITIVITIES          *
C     *       ACCUR   ARRAY OF M COMPONENTS REPRESENTING THE ACCURACY     *
C     *               OF THE RECORDING SYSTEM, USED TO ESTABLISH THE      *
C     *               INTERACTION CONVERGENCE CRITERIA. IT IS ASSUMED     *
C     *               THAT ALL ELEMENTS OF ACCUR ARE GREATER THAN 0.      *
C     *                                                                   *
C     *   BALANCE INTERACTION HISTORY FILE                                *
C     *      THE INPUT COMMON PARAMETERS RESIDE ON A BALANCE              *
C     *      INTERACTION HISTORY FILE.  THE FILE CONSISTS OF A            *
C     *      PAIR OF RECORDS FOR EACH BALANCE.  THE FIRST RECORD          *
C     *      OF EACH PAIR CONTAINS THE ORIGINAL CALIBRATION MATRIX        *
C     *      C(162).  THIS SUBROUTINE USES THE SECOND RECORD,             *
C     *      WHICH CONTAINS THE INVERSELY DERIVED MATRICES C1I(36)        *
C     *      AND C1IC2(126)                                               *
C     *                                                                   *
C     *   REMARKS                                                         *
C     *      THIS SUBROUTINE IS DESIGNED IN SUCH A WAY THAT ALL           *
C     *      COMPONENT AND CALIBRATION MATRICES COULD BE COLLAPSED        *
C     *      TO ONLY THOSE COMPONENTS THAT ARE ACTUALLY HOOKED UP         *
C     *                                                                   *
C     *   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   *
C     *      GMEQF   EQUATES TWO MATRICES                                 *
C     *      GMADF   ADDS TWO MATRICES                                    *
C     *      GMSBF   SUBTRACTS TWO MATRICES                               *
C     *      GMPXF   MULTIPLIES TWO MATRICES                              *
C     *      TESTF   COMPARES TWO MATRICES                                *
C     *      GMSTF   SETS A MATRIX EQUAL TO A SCALAR                      *
C     *      MATAS   MULTIPLIES A MATRIX BY ITS TRANSPOSE AND STORES      *
C     *              THE UPPER TRIANGLE OF THE PRODUCT 1-DIMENSIONALLY    *
C     *                                                                   *
C     *   METHOD                                                          *
C     *      DETERMINING CORRECT COMPONENTS F IS VIEWED AS AN             *
C     *      ITERATIVE SOLUTION TO THE FOLLOWING MATRIX EQUATION          *
C     *                                                                   *
C     *                    F = C1I X FU - C1IC2 X F2                      *
C     *                  (M,1) (M,M) (M,1)  (M,N) (N,1)                   *
C     *                                                                   *
C     *      WHERE F2 IS ALL PRODUCT COMBINATIONS OF F AND N=M(M+1)/2     *
C     *      LETTING EPSI = C1IC2 X F2  THE ITERATIVE TECHNIQUE FOLLOWS   *
C     *                                                                   *
C     *                APPROXIMATION          RESULTS          ERROR      *
C     *                                                                   *
C     *          F = C1I X FU          = C1I X FU - EPS1     EPS1         *
C     *          F = C1I X FU - EPS1   = C1I X FU - EPS2     EPS2-EPS1    *
C     *          F = C1I X FU - EPS2   = C1I X FU - EPS3     EPS3-EPS2    *
C     *          .                .              .               .        *
C     *          .                .              .               .        *
C     *          .                .              .               .        *
C     *          UNTIL ABSOLUTE(EPS(I)-EPS(I-1)) < ABSOLUTE(ACCUR)        *
C     *                                                                   *
C     *                          FOR ALL COMPONENTS                       *
C     *                                                                   *
C     *********************************************************************
C
C
      DIMENSION FU(6),FZ(6),EZ(6),F(6)
      DATA IZ/0/
```

```
C
C
C     ..................................................................
C     .                                                                .
C     .INPUTS FROM THE BALANCE INTERACTION HISTORY FILE                .
C     .                                                                .
C     ..................................................................
C
      COMMON/BAL/
     1IBAL,ICATE,KCATE,M,NAMEC(6),ITASK,IORDR,ITRNL,IPLUS,MINUS,NTRY,
     2C1I(36),C1IC2(126),CPOS(6),CNEG(6),PRCNT,CSENS(6),ACCUR(6)
C
C
C     ..................................................................
C     .                                                                .
C     .WORKING STORAGE AREA AVAILABLE TO ALL SUBPROGRAMS                .
C     .                                                                .
C     ..................................................................
C
      COMMON/WORK/
     1N,F2(21),EPSI(6),EPSO(6),DELTA(6),I,J,ICNVG
C
C
C     ..................................................................
C     .                                                                .
C     .NO INTERACTIONS, NOT NECESSARY TO CORRECT COMPONENTS             .
C     .                                                                .
C     ..................................................................
C
      IER=0
      IF(IORDR) 20,10,20
C
C
C          ..............................................................
C          .                                                            .
C          .SET CORRECTED EQUAL TO UNCORRECTED AND RETURN               .
C          .                                                            .
C          ..............................................................
C
   10      CALL GMEQF(FU,F,M)
           GO TO 200
C
C
C     ..................................................................
C     .                                                                .
C     .CORRECT COMPONENTS FOR 1ST ORDER INTERACTIONS EXPLICITLY         .
C     .                                                                .
C     ..................................................................
C
   20 CALL GMPXF(C1I,FU,F,M,M,1)
C
C
C          ..............................................................
C          .                                                            .
C          .THIS IS THE FIRST APPROXIMATION FOR 2ND ORDER CORRECTIONS    .
C          .                                                            .
C          ..............................................................
C
           IF(ICRDR-1) 30,200,30
C
C
```

```
C     ..................................................................
C     .                                                                .
C     .OPTION TO TRANSLATE AXES, COMPENSATING FOR INITIAL LOADS        .
C     .                                                                .
C     ..................................................................
C
   30 IF(ITRNL) 50,40,50
C
C
C          ..............................................................
C          .                                                            .
C          .TRANSLATION NOT NECESSARY, INITIALIZE EPSILON TO ZERO       .
C          .                                                            .
C          ..............................................................
C
   40       CALL GMSTF(EPS0,0.,M)
            GO TO 60
C
C
C          ..............................................................
C          .                                                            .
C          .TRANSLATION NECESSARY,ADD INITIAL LOADS TO 1ST APPROXIMATION.
C          .                                                            .
C          ..............................................................
C
   50       CALL GMADF(F,FZ,F,M)
C
C
C          ..............................................................
C          .                                                            .
C          .INITIALIZE EPSILON TO 2ND ORDER INTERACTION ON INITIAL LOADS.
C          .                                                            .
C          ..............................................................
C
            CALL GMEQF(EZ,EPS0,M)
C
C
C     ..................................................................
C     .                                                                .
C     .OPTION TO HANDLE ONE DISCONTINUOUS 2ND ORDER INTERACTION TERM    .
C     .                                                                .
C     ..................................................................
C
   60 IF(IPLUS) 70,100,70
C
C
C          ..............................................................
C          .                                                            .
C          .DETERMINE WHETHER TO USE POSITIVE OR NEGATIVE CALIBRATION   .
C          .                                                            .
C          ..............................................................
C
   70       IF(F(IPLUS)) 80,90,90
C
C
C               .........................................................
C               .                                                       .
C               .SETUP TO USE INTERACTION TERM FROM NEGATIVE CALIBARTION.
C               .                                                       .
C               .........................................................
```

```
C
   80            CALL GMEQF(CNEG,C1IC2(MINUS),M)
                 GO TO 100
C
C
C
C
C
C              ..........................................................
C              .                                                        .
C              .SETUP TO USE INTERACTION TERM FROM POSITIVE CALIBRATION.
C              .                                                        .
C              ..........................................................
C
   90            CALL GMEQF(CPOS,C1IC2(MINUS),M)
C
C
C     ..................................................................
C     .                                                                .
C     .PRINT OPTION TO DISPLAY THE INTERACTION CONVERGENCE PATTERN      .
C     .                                                                .
C     ..................................................................
C
  100 IF(LIST) 110,130,110
C
C
C          ..............................................................
C          .                                                            .
C          .ESTABLISH HEADINGS OF ITERATION AND COMPONENT NAMES         .
C          .                                                            .
C          ..............................................................
C
  110       WRITE(LIST,111) (NAMEC(J),J=1,M)
  111       FORMAT(10HOITERATION,8X,A2,5(17X,A2))
C
C
C          ..............................................................
C          .                                                            .
C          .PRINT FIRST APPROXIMATION DEEMED ITERATION NUMBER 0         .
C          .                                                            .
C          ..............................................................
C
            WRITE(LIST,120) IZ,(F(J),J=1,M)
  120       FORMAT(I6,6E19.6)
C
C
C     ..................................................................
C     .                                                                .
C     .CORRECT COMPONENTS FOR 2ND ORDER INTERACTIONS ITERATIVELY        .
C     .                                                                .
C     ..................................................................
C
  130 DO 170 I=1,NTRY
C
C
C          ..............................................................
C          .                                                            .
C          .COMPUTE ALL PRODUCT COMBINATIONS OF THIS APPROXIMATION       .
C          .                                                            .
C          ..............................................................
```

```
C
      CALL MATAS(F,M,F2,N)
C
C
C     ..................................................................
C     .                                                                .
C     .COMPLTE 2ND ORDER INTERACTION DUE TO THIS APPROXIMATION         .
C     .                                                                .
C     ..................................................................
C
      CALL CMPXF(C1IC2,F2,EPSI,M,N,1)
C
C
C     ..................................................................
C     .                                                                .
C     .COMPLTE ERROR IN THIS APPROXIMATION OF CORRECTED COMPONENTS      .
C     .                                                                .
C     ..................................................................
C
      CALL CMSBF(EPSI,EPSO,DELTA,M)
C
C
C     ..................................................................
C     .                                                                .
C     .COMPLTE NEXT APPROXIMATION OF THE CORRECTED COMPONENTS           .
C     .                                                                .
C     ..................................................................
C
      CALL CMSBF(F,DELTA,F,M)
C
C
C     ..................................................................
C     .                                                                .
C     .PRINT CPTION TO DISPLAY TOTAL LOADS PER ITERATION               .
C     .                                                                .
C     ..................................................................
C
      IF(LIST) 140,150,140
  140 WRITE(LIST,120) I,(F(J),J=1,M)
C
C
C     ..................................................................
C     .                                                                .
C     .DEMAND SIMULTANEOUS CONVERGENCE OF ALL COMPONENTS               .
C     .                                                                .
C     ..................................................................
C
  150 CALL TESTF(DELTA,ACCUR,M,ICNVG)
C
C
C     ..................................................................
C     .                                                                .
C     .DID INTERACTIONS CONVERGE TO WITHIN A PRESCRIBED ACCURACY        .
C     .                                                                .
C     ..................................................................
C
      IF(ICNVG) 180,160,180
C
C
C         ..............................................................
C         .                                                            .
C         .NO, SAVE RESULTS OF THIS ITERATION AND TRY AGAIN OR         .
C         .                                                            .
C         ..............................................................
```

```
C
  160            CALL GMEQF(EPSI,EPSO,M)
C
C
C               .....................................................
C               .                                                   .
C               .IF MAXIMUM TRYS EXCEEDED, SET ERROR FLAG AND RETURN .
C               .                                                   .
C               .....................................................
C
  170            CONTINUE
                 IER=1
C
C
C    .....................................................................
C    .                                                                   .
C    .TRANSLATE AXES BACK TO ORIGINAL SYSTEM                             .
C    .                                                                   .
C    .....................................................................
C
  180 IF(ITRNL) 190,200,190
C
C
C         .....................................................
C         .                                                   .
C         .SUBTRACT INITIAL LOADS FROM TOTAL LOADS            .
C         .                                                   .
C         .....................................................
C
  190      CALL GMSBF(F,FZ,F,M)
C
C
C         .....................................................
C         .                                                   .
C         .RETURN COMPONENTS CORRECTED FOR INTERACTIONS       .
C         .                                                   .
C         .....................................................
C
  200      RETURN
C
C
      END
```

```
      SUBROUTINE CTRNL(FZ,EZ)
C
C     ******************************************************************
C     *                                                                *
C     *   SUBROUTINE                                                   *
C     *      CTRNL                                                     *
C     *                                                                *
C     *   PURPOSE                                                      *
C     *      COMPUTE 2ND ORDER INTERACTION DUE TO INITIAL LOADS.       *
C     *      PROVIDE INPUT TO SUBROUTINE CINTR FOR AXES TRANSLATION    *
C     *                                                                *
C     *   LANGUAGE                                                     *
C     *      FORTRAN 2 OR 4                                            *
C     *                                                                *
C     *   USAGE                                                        *
C     *      DEFINE INPUT COMMON PARAMETERS                            *
C     *      CALL CTRNL(FZ,EZ)                                         *
C     *                                                                *
C     *   DESCRIPTION OF INPUT CALLING SEQUENCE PARAMETER              *
C     *      FZ     CORRECT INITIAL LOADS, DETERMINED ITERATIVELY       *
C     *                                                                *
C     *   DESCRIPTION OF OUTPUT CALLING SEQUENCE PARAMETER             *
C     *      EZ     2ND ORDER INTERACTION DUE TO CORRECT INITIAL LOADS  *
C     *                                                                *
C     *   REMARKS                                                      *
C     *      THE INPUT COMMON PARAMETERS RESIDE ON A BALANCE INTERACTION*
C     *      HISTORY FILE.  SEE SUBROUTINE CINTR FOR A DESCRIPTION OF  *
C     *      THESE PARAMETERS                                          *
C     *                                                                *
C     *   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                *
C     *      GMEQF  EQUATES TWO MATRICES                               *
C     *      GMPXF  MULTIPLIES TWO MATRICES                            *
C     *      GMSTF  SETS A MATRIX EQUAL TO A SCALAR                    *
C     *      MATAS  MULTIPLIES A MATRIX BY ITS TRANSPOSE AND STORES    *
C     *             THE UPPER TRIANGLE IN 1-DIMENSIONAL SYMMETRIC FORM *
C     *                                                                *
C     *   METHOD                                                       *
C     *                                                                *
C     *             FZ = C1I X FUZ - C1IC2 X F2Z                       *
C     *             WHERE UNCORRECTED INITIAL LOADS FUZ ARE NEVER      *
C     *             REALLY KNOWN AND F2Z IS ALL PRODUCT COMBINATIONS   *
C     *             OF FZ. FOR THE PURPOSE OF TRANSLATING AXES IN      *
C     *             SUBROUTINE CINTR IT IS SUFFICIENT TO KNOW FZ AND   *
C     *             EZ = C1IC2 X F2Z  BY DEFINITION                    *
C     *                                                                *
C     ******************************************************************
C
C
      DIMENSION FZ(6),EZ(6)
C
C
C     ..................................................................
C     .                                                                .
C     .INPUTS FROM THE BALANCE INTERACTION HISTORY FILE                .
C     .                                                                .
C     ..................................................................
C
      COMMON/BAL/
     1IBAL,IDATE,KDATE,M,NAMEC(6),ITASK,IORDR,ITRNL,IPLUS,MINUS,NTRY,
     2C1I(36),C1IC2(126),CPOS(6),CNEG(6),PRCNT,CSENS(6),ACCUR(6)
C
C
```

```
C     ..................................................................
C     .                                                                .
C     .WORKING STORAGE AREA AVAILABLE TO ALL SUBPROGRAMS                .
C     .                                                                .
C     ..................................................................
C
      COMMON/WORK/N,F2Z(21)
C
C
C     ..................................................................
C     .                                                                .
C     .TRANSLATION ONLY NECESSARY FOR 2ND ORDER INTERACTIONS            .
C     .                                                                .
C     ..................................................................
C
      IF(IORDR-2) 10,20,10
C
C
C          ..............................................................
C          .                                                            .
C          .OTHERWISE, SET INITIAL EPSILON TO ZERO                      .
C          .                                                            .
C          ..............................................................
C
   10      CALL GMSTF(EZ,0.,M)
           GO TO 100
C
C
C     ..................................................................
C     .                                                                .
C     .OPTION TO HANDLE ONE DISCONTINUOUS 2ND ORDER INTERACTION TERM    .
C     .                                                                .
C     ..................................................................
C
   20 IF(IPLUS) 30,60,30
C
C
C          ..............................................................
C          .                                                            .
C          .DETERMINE WHETHER TO USE POSITIVE OR NEGATIVE CALIBRATION   .
C          .                                                            .
C          ..............................................................
C
   30      IF(FZ(IPLUS)) 40,50,50
C
C
C               ........................................................
C               .                                                      .
C               .SETUP TO USE INTERACTION TERM FROM NEGATIVE CALIBRATION.
C               .                                                      .
C               ........................................................
C
   40           CALL GMEQF(CNEG,C1IC2(MINUS),M)
                GO TO 60
C
C
C               ........................................................
C               .                                                      .
C               .SETUP TO USE INTERACTION TERM FROM POSITIVE CALIBRATION.
C               .                                                      .
C               ........................................................
C
   50           CALL GMEQF(CPOS,C1IC2(MINUS),M)
C
C
C
```

```
C     ..................................................................
C     .                                                                .
C     .COMPUTE ALL PRODUCT COMBINATIONS OF CORRECT INITIAL LOADS        .
C     .                                                                .
C     ..................................................................
C
   60 CALL MATAS(FZ,M,F2Z,N)
C
C
C     ..................................................................
C     .                                                                .
C     .COMPUTE 2ND ORDER INTERACTION DUE TO CORRECT INITIAL LOADS       .
C     .                                                                .
C     ..................................................................
C
      CALL GMPXF(C1IC2,F2Z,EZ,M,N,1)
C
C
  100 RETURN
      END

      SUBROUTINE GMEQF(A,R,MN)
C
C     ***********************************************************************
C     *                                                                     *
C     *   SUBROUTINE                                                        *
C     *      GMEQF                                                          *
C     *                                                                     *
C     *   PURPOSE                                                           *
C     *      EQUATE ONE GENERAL MATRIX TO ANOTHER GENERAL MATRIX            *
C     *                                                                     *
C     *   LANGUAGE                                                          *
C     *      FORTRAN 2 OR 4                                                 *
C     *                                                                     *
C     *   USAGE                                                             *
C     *      CALL GMEQF(A,R,MN)                                             *
C     *                                                                     *
C     *   DESCRIPTION OF PARAMETERS                                         *
C     *      A       INPUT MATRIX NAME                                      *
C     *      R       OUTPUT MATRIX NAME                                     *
C     *      MN      INPUT NUMBER OF ELEMENTS IN MATRIX A OR R              *
C     *                                                                     *
C     *   REMARKS                                                           *
C     *   1. THE ELEMENTS OF MATRIX A ARE NOT CHANGED.                      *
C     *   2. THE USER IS CAUTIONED,IF MATRICES A AND R ARE NOT FLOATING     *
C     *      POINT. FOR EXAMPLE, TWO INTEGER TO ONE FLOATING POINT WORD.    *
C     *   3. SUBROUTINE GMEQF CAN BE USED TO MANIPULATE MATRIX COLUMNS.     *
C     *      FOR EXAMPLE, SET MATRIX R(M,1) EQUAL TO THE JTH COLUMN OF      *
C     *      MATRIX A(M,N) BY CALL GMEQF(A(1,J),R,M).                       *
C     *   4. SUBROUTINE GMEQF CANNOT BE EASILY USED TO MANIPULATE ROWS      *
C     *      DUE TO THE FACT THAT THE ELEMENTS OF A ROW ARE NOT HELD        *
C     *      CONSECUTIVELY IN CORE STORAGE.                                 *
C     *                                                                     *
C     *   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     *
C     *      NONE                                                           *
C     *                                                                     *
C     *   METHOD                                                            *
C     *      EACH ELEMENT OF MATRIX R IS SET EQUAL TO                       *
C     *      THE CORRESPONDING ELEMENT OF MATRIX A                          *
C     *                                                                     *
C     *            R(IJ)=A(IJ) FOR IJ=1,2,...,MN                            *
C     *                                                                     *
C     ***********************************************************************
```

```
C
C        EQUATE MATRICES
C
      DIMENSION A(1),R(1)
      DO 10 IJ=1,MN
      R(IJ)=A(IJ)
   10 CONTINUE
      RETURN
      END
      SUBROUTINE GMADF(A,B,R,MN)
C
C     ***************************************************************************
C     *                                                                         *
C     *   SUBROUTINE                                                            *
C     *      GMADF                                                              *
C     *                                                                         *
C     *   PURPOSE                                                               *
C     *      ADD TWO GENERAL MATRICES TO FORM RESULTANT GENERAL MATRIX          *
C     *                                                                         *
C     *   LANGUAGE                                                              *
C     *      FORTRAN 2 OR 4                                                     *
C     *                                                                         *
C     *   USAGE                                                                 *
C     *      CALL GMADF(A,B,R,MN)                                               *
C     *                                                                         *
C     *   DESCRIPTION OF PARAMETERS                                             *
C     *      A       INPUT FIRST MATRIX NAME                                    *
C     *      B       INPUT SECOND MATRIX NAME                                   *
C     *      R       OUTPUT MATRIX NAME                                         *
C     *      MN      INPUT NUMBER OF ELEMENTS IN MATRIX A,B,OR R                *
C     *                                                                         *
C     *   REMARKS                                                               *
C     *      MATRICES A,B, AND R MUST BE FLOATING POINT                         *
C     *      MATRICES A,B, AND R MAY BE THE SAME LOCATIONS                      *
C     *      OTHERWISE, THE ELEMENTS OF MATRICES A,B ARE NOT CHANGED            *
C     *                                                                         *
C     *   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                         *
C     *      NONE                                                               *
C     *                                                                         *
C     *   METHOD                                                                *
C     *      EACH ELEMENT OF MATRIX A IS ADDED TO THE CORRESPONDING             *
C     *      ELEMENT OF MATRIX B AND THE RESULT IS PLACED IN THE                *
C     *      CORRESPONDING ELEMENT OF MATRIX R                                  *
C     *                                                                         *
C     *            R(IJ)=A(IJ)+B(IJ) FOR IJ=1,2,...,MN                          *
C     *                                                                         *
C     ***************************************************************************
C
C        ADD MATRICES
C
      DIMENSION A(1),B(1),R(1)
      DO 10 IJ=1,MN
      R(IJ)=A(IJ)+B(IJ)
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE GMSBF(A,B,R,MN)
C
C     ******************************************************************
C     *                                                                *
C     *   SUBROUTINE                                                   *
C     *      GMSBF                                                     *
C     *                                                                *
C     *   PURPOSE                                                      *
C     *      SUBTRACT ONE GENERAL MATRIX FROM ANOTHER                  *
C     *      TO FORM A RESULTANT GENERAL MATRIX                        *
C     *                                                                *
C     *   LANGUAGE                                                     *
C     *      FORTRAN 2 OR 4                                            *
C     *                                                                *
C     *   USAGE                                                        *
C     *      CALL GMSBF(A,B,R,MN)                                      *
C     *                                                                *
C     *   DESCRIPTION OF PARAMETERS                                    *
C     *      A       INPUT NAME OF FIRST MATRIX                        *
C     *      B       INPUT NAME OF SECOND MATRIX                       *
C     *      R       OUTPUT MATRIX NAME                                *
C     *      MN      INPUT NUMBER OF ELEMENTS IN MATRIX A,B,OR R       *
C     *                                                                *
C     *   REMARKS                                                      *
C     *      MATRICES A,B, AND R MUST BE FLOATING POINT                *
C     *      MATRICES A,B, AND R MAY BE THE SAME LOCATIONS             *
C     *      OTHERWISE, THE ELEMENTS OF MATRICES A,B ARE NOT CHANGED   *
C     *                                                                *
C     *   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                *
C     *      NONE                                                      *
C     *                                                                *
C     *   METHOD                                                       *
C     *      EACH ELEMENT OF MATRIX B IS SUBTRACTED FROM THE           *
C     *      CORRESPONDING ELEMENT OF MATRIX A AND THE RESULT          *
C     *      IS PLACED IN THE CORRESPONDING ELEMENT OF MATRIX R        *
C     *                                                                *
C     *                  R(IJ)=A(IJ)-B(IJ) FOR IJ=1,2,...,MN           *
C     *                                                                *
C     ******************************************************************
C
C         SUBTRACT MATRICES
C
            DIMENSION A(1),B(1),R(1)
            DO 10 IJ=1,MN
            R(IJ)=A(IJ)-B(IJ)
         10 CONTINUE
            RETURN
            END
```

```
      SUBROUTINE GMPXF(A,B,R,M,N,L)
C
C     ************************************************************************
C     *                                                                      *
C     *   SUBROUTINE                                                         *
C     *      GMPXF                                                           *
C     *                                                                      *
C     *   PURPOSE                                                            *
C     *      MULTIPLY TWO GENERAL MATRICES                                   *
C     *      TO FORM A RESULTANT GENERAL MATRIX                              *
C     *                                                                      *
C     *   LANGUAGE                                                           *
C     *      FORTRAN 2 OR 4                                                  *
C     *                                                                      *
C     *   USAGE                                                              *
C     *      CALL GMPXF(A,B,R,M,N,L)                                         *
C     *                                                                      *
C     *   DESCRIPTION OF PARAMETERS                                          *
C     *      A       INPUT FIRST MATRIX NAME                                 *
C     *      B       INPUT SECOND MATRIX NAME                                *
C     *      R       OUTPUT MATRIX NAME                                      *
C     *      M       INPUT NUMBER OF ROWS IN MATRIX A OR R                   *
C     *      N       INPUT NUMBER OF COLUMNS IN A AND ROWS IN B              *
C     *      L       INPUT NUMBER OF COLUMNS IN MATRIX B OR R                *
C     *                                                                      *
C     *   REMARKS                                                            *
C     *      ALL MATRICES MUST BE STORED IN FLOATING POINT                   *
C     *      A AND B MUST BE CONFORMABLE FOR MATRIX MULTIPLICATION           *
C     *      A AND B MAY BE THE SAME MATRIX IF IT IS SQUARE                  *
C     *      MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A OR B        *
C     *      THE ELEMENTS OF MATRICES A AND B ARE NOT CHANGED                *
C     *                                                                      *
C     *   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                      *
C     *      NONE                                                            *
C     *                                                                      *
C     *   METHOD                                                             *
C     *      THE M BY N MATRIX A IS POSTMULTIPLIED BY THE N BY L             *
C     *      MATRIX B AND THE RESULT IS STORED IN THE M BY L MATRIX R.       *
C     *                                                                      *
C     *              FOR A GIVEN ROW I AND COLUMN J,                         *
C     *              R(I,J)=THE SUMMATION FROM K=1,2,...,N                   *
C     *              OF THE PRODUCTS A(I,K)*B(K,J)                           *
C     *                                                                      *
C     ************************************************************************
C
C        MULTIPLY MATRICES
C
      DIMENSION A(1),B(1),R(1)
      IR=0
      IK=-N
      DO 30 K=1,L
      IK=IK+N
      DO 20 J=1,M
      IR=IR+1
      JI=J-M
      IB=IK
      R(IR)=0.
      DO 10 I=1,N
      JI=JI+M
      IB=IB+1
      R(IR)=R(IR)+A(JI)*B(IB)
   10 CONTINUE
   20 CONTINUE
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TESTF(A,B,MN,LE)
C
C     ******************************************************************
C     *                                                                *
C     *   SUBROUTINE                                                   *
C     *      TESTF                                                     *
C     *                                                                *
C     *   PURPOSE                                                      *
C     *      TEST THE ABSOLUTE VALUE OF EACH ELEMENT OF MATRIX A TO    *
C     *      DETERMINE IF IT IS LESS THAN OR EQUAL TO THE CORRESPONDING *
C     *      ELEMENT OF MATRIX B                                       *
C     *                                                                *
C     *   LANGUAGE                                                     *
C     *      FORTRAN 2 OR 4                                            *
C     *                                                                *
C     *   USAGE                                                        *
C     *      CALL TESTF(A,B,MN,LE)                                     *
C     *                                                                *
C     *   DESCRIPTION OF PARAMETERS                                    *
C     *      A       INPUT FIRST MATRIX NAME                           *
C     *      B       INPUT SECOND MATRIX NAME                          *
C     *      MN      INPUT NUMBER OF ELEMENTS IN MATRIX A OR B         *
C     *      LE      OUTPUT COMPARISON OF MATRICES A AND B             *
C     *                                                                *
C     *   REMARKS                                                      *
C     *      LE=0 IF THE ABSOLUTE VALUE OF AT LEAST ONE ELEMENT IN     *
C     *      MATRIX A IS GREATER THAN THE VALUE OF THE CORRESPONDING   *
C     *      ELEMENT IN MATRIX B.  OTHERWISE, LE=1                     *
C     *                                                                *
C     *   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                *
C     *      NONE                                                      *
C     *                                                                *
C     *   METHOD                                                       *
C     *      IF |A(IJ)| LESS THAN OR EQUAL TO B(IJ)                    *
C     *      FOR ALL IJ=1,2,...,MN THEN LE=1                           *
C     *      OTHERWISE, LE=0                                           *
C     *                                                                *
C     ******************************************************************
C
C        COMPARE MATRICES
C
      DIMENSION A(1),B(1)
      LE=0
      DO 10 IJ=1,MN
      IF(ABS(A(IJ))-B(IJ)) 10,10,20
   10 CONTINUE
      LE=1
   20 RETURN
      END
```

```
      SUBROUTINE GMSTF(R,S,MN)
C
C     ***************************************************************
C     *                                                             *
C     *   SUBROUTINE                                                *
C     *      GMSTF                                                  *
C     *                                                             *
C     *   PURPOSE                                                   *
C     *      SET ALL ELEMENTS OF A GENERAL MATRIX EQUAL TO A SCALAR *
C     *                                                             *
C     *   LANGUAGE                                                  *
C     *      FORTRAN 2 OR 4                                         *
C     *                                                             *
C     *   USAGE                                                     *
C     *      CALL GMSTF(R,S,MN)                                     *
C     *                                                             *
C     *   DESCRIPTION OF PARAMETERS                                 *
C     *      R       OUTPUT MATRIX NAME                             *
C     *      S       INPUT SCALAR CONSTANT                          *
C     *      MN      INPUT NUMBER OF ELEMENTS IN MATRIX R           *
C     *                                                             *
C     *   REMARKS                                                   *
C     *      ALL VARIABLES SHOULD BE FLOATING POINT                 *
C     *                                                             *
C     *   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED             *
C     *      NONE                                                   *
C     *                                                             *
C     *   METHOD                                                    *
C     *      SET EACH ELEMENT OF MATRIX R EQUAL TO THE SCALAR S     *
C     *                                                             *
C     *              R(IJ)=S FOR ALL IJ=1,2,...,MN                  *
C     *                                                             *
C     ***************************************************************
C
C        SET EACH ELEMENT OF MATRIX R EQUAL TO THE SCALAR S
C
      DIMENSION R(1)
      DO 10 IJ=1,MN
      R(IJ)=S
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MATAS(A,M,R,N)
C
C     ***************************************************************
C     *                                                             *
C     *   PURPOSE                                                   *
C     *      POSTMULTIPLY A COLUMN MATRIX BY ITS TRANSPOSE AND STORE THE*
C     *      UPPER TRIANGLE OF THE RESULTANT MATRIX IN SYMMETRIC FORM  *
C     *                                                             *
C     *   LANGUAGE                                                  *
C     *      FORTRAN 2 OR 4                                         *
C     *                                                             *
C     *   USAGE                                                     *
C     *      CALL MATAS(A,M,R,N)                                    *
C     *                                                             *
C     *   DESCRIPTION OF PARAMETERS                                 *
C     *      A       INPUT MATRIX NAME                              *
C     *      M       INPUT NUMBER OF ELEMENTS IN MATRIX A           *
C     *      R       OUTPUT MATRIX NAME                             *
C     *      N       OUTPUT NUMBER OF ELEMENTS IN MATRIX R          *
C     *                                                             *
C     *   EXAMPLE |-1   |-          -1   |-               -1   |--1     *
C     *           |N| X |N A P R Y S| = |NN NA NP NR NY NS| = |NN| = R  *
C     *           |A|   |-          -1   |  AA AP AR AY AS|   |NA|      *
C     *           |P|         (1,6)      |     PP PR PY PS|   |NP|      *
C     *           |R|                    |        RR RY RS|   |NR|      *
C     *           |Y|                    |           YY YS|   |NY|      *
C     *           |S|                    |              SS|   |NS|      *
C     *           |-|           |-1      |                -|   |AA|      *
C     *         (6,1)           |N|      |             (6,6)   |AP|      *
C     *                         |A|      |                     |AR|      *
C     *                 WHERE A = |P|    |                     |AY|      *
C     *                         |R|      |                     |AS|      *
C     *                         |Y|      |                     |PP|      *
C     *                         |S|      |                     |PR|      *
C     *                         |-|      |                     |PY|      *
C     *                                  |                     |PS|      *
C     *                       M=6,N=21   |                     |RR|      *
C     *                                  |                     |RY|      *
C     *                                  |                     |RS|      *
C     *                                  |                     |YY|      *
C     *                                  |                     |YS|      *
C     *                                  |                     |SS|      *
C     *                                  |                    |--1      *
C     *                                  |                  (21,1)      *
C     *                                                             *
C     *   REMARKS                                                   *
C     *   1. THE RESULTANT NUMBER OF ELEMENTS IN MATRIX R IS N=M(M+1)/2 *
C     *   2. FOR COMPUTER EFFICIENCY, MATRIX A IS RESTRICTED TO 1 COLUMN*
C     *   3. MATRICES A AND R CANNOT SHARE THE SAME LOCATIONS       *
C     *   4. THE ELEMENTS OF MATRIX A ARE NOT CHANGED               *
C     *   5. MATRIX R REPRESENTS ALL PRODUCT COMBINATIONS OF M ELEMENTS *
C     *                                                             *
C     *   FUNCTIONS AND SUBPROGRAMS REQUIRED                        *
C     *      NONE                                                   *
C     *                                                             *
C     *   METHOD                                                    *
C     *      ANY MATRIX A(M,L) TIMES ITS TRANSPOSE AT(L,M) RESULTS IN A *
C     *      SYMMETRIC MATRIX R(M,M). THIS SUBROUTINE HAS RESTRICTED   *
C     *      L TO 1. THE UPPER AND LOWER TRIANGLES OF MATRIX R ARE     *
C     *      IMAGES OF ONE ANOTHER. CERTAIN APPLICATIONS REQUIRE       *
C     *      ONLY THE UPPER OR LOWER TRIANGLE STORED 1-DIMENSIONALLY.  *
C     *      THE ABOVE EXAMPLE DEMONSTRATES THE 1-DIMENSIONAL ORDERING *
C     *                                                             *
C     ***************************************************************
```

```
C
C         PERFORM THE MATRIX OPERATION
C
      DIMENSION A(1),R(1)
      N=0
      DO 20 I=1,M
      DO 10 J=I,M
      N=N+1
      R(N)=A(I)*A(J)
   10 CONTINUE
   20 CONTINUE
      RETURN
      END
```

REFERENCES

1. Hansen, Raymond M.: Evaluation and Calibration of Wire-Strain-Gage Wind-Tunnel Balances Under Load. AGARD Rep. 13, Feb. 1956.

2. Cook, T. A.: A Note on the Calibration of Strain Gauge Balances for Wind Tunnel Models. Tech. Note No. Aero. 2631, Brit. R.A.E., Dec. 1959.

3. Faddeeva, V. N. (Curtis D. Benster, trans.): Computational Methods of Linear Algebra. Dover Publ., Inc., c.1959.

4. Henrici, Peter: Elements of Numerical Analysis. John Wiley & Sons, Inc., c.1964.

5. Zosel, Mary E.: BSWT Force Data, Sting Mount. Doc. No. D6-29487, Boeing Co., 1968.

6. Sokolnikoff, I. S.; and Redheffer, R. M.: Mathematics of Physics and Modern Engineering. Second ed., McGraw-Hill Book Co., Inc., c.1966.

NASA-Langley, 1972 — 1 L-8278

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

— NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

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

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

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

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

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

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

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

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

SCIENTIFIC AND TECHNICAL INFORMATION OFFICE

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546