6647.7-M
(1)
702047

SPARSE MATRIX TECHNIQUES IN TWO MATHEMATICAL PROGRAMMING CODES

BY

George B. Dantzig
Roy P. Harvey
Robert D. McKnight
Stanley S. Smith

TECHNICAL REPORT NO. 69-1
JANUARY 1969

This document has been approved for public release and sale; its distribution is unlimited.

OPERATIONS RESEARCH HOUSE

DDC
RECEIVED
MAR 11 1970
C

Stanford University
CALIFORNIA

Reproduced by the
CLEARINGHOUSE
for Federal Scientific & Technical
Information Springfield Va. 22151

SPARSE MATRIX TECHNIQUES IN TWO MATHEMATICAL PROGRAMMING CODES

by

George B. Dantzig
Roy P. Harvey†
Robert D. McKnight†
Stanley S. Smith†

Technical Report No. 69-1
January 1969

Operations Research House
Stanford University
Stanford, California

† Standard Oil of California

Research of George B. Dantzig and reproduction of this report was partially supported by Office of Naval Research, Contracts ONR-N-00014-67-A-0112-0011 and ONR-N-00014-67-A-0112-0016; U.S. Atomic Energy Commission, Contract AT[04-3]-326 PA #18; National Science Foundation Grant GP 6431; and U.S. Army Research Office, Contract DAHC04-67-C0028.

Reproduction in whole or in part for any purpose of the United States Government is permitted.

SPARSE MATRIX TECHNIQUES IN TWO MATHEMATICAL PROGRAMMING CODES

by

George B. Dantzig*
Roy P. Harvey†
Robert D. McKnight†
Stanley S. Smith†

Introduction. The M3 and M5 linear/separable programming codes [1] and [2] were written for the IBM 7090/94 class of computer using the FORTRAN Monitor System. They solve problems up to about 400 rows. M3 was developed first, during the period 1960-62, in part by RAND Corporation personnel and in part by the Standard Oil of California (SOCAL). It has been available through the SHARE library for several years. M5 was developed later, during the period 1962-64, by the authors of this paper and is a SOCAL proprietary code.

Through the use of sparse matrix techniques and other devices, M5 can solve a problem in about one quarter of the time taken by M3. Before discussing these improvements, we list first some of the features that the codes have in common:

- Single Precision
- Upper Bounding Algorithm [3]
- Cost Ranging Algorithm [4], [5]
- Composite Algorithm [6]
- An Upper-Bounded and a Non-Upper-Bounded Separable-Programming Algorithm [7], [8], [9], [10], [11]
- The ability to designate free variables by use of controls (a free variable is unrestricted as regards sign)
- The ability to designate frozen variables by use of controls (a frozen variable must be zero in the final solution whether in or out of the basis)

* Stanford University, Stanford, California
† Standard Oil of California

- Getoff and Restart Procedures
- The ability to handle multiple objectives and right hand sides without the need to recalculate the inverse
- Row and Column Error Calculations as desired
- An Automatic Tolerance Regulation Mode of Operation [12]
- A Pivot Rejection Algorithm
- Crashing Procedures
- Multiple Pricing: The selection (and possible use) of several attractive non-basic columns during the pricing-out operation in a simplex iteration

The most interesting aspects of the codes with respect to the topic of sparse matrix methods are in connection with the inversion techniques. The search for more and more efficient methods for computer solution of large linear programming problems has resulted in procedures that for the most part are really only variants of the simplex method. Some of these variants, such as decomposition, represent fairly radical departures; others differ only in computational form. The so-called revised simplex method is perhaps an example of the latter group [13].

In review, recall that the "tableau" of the original simplex method, is, in effect, the $m \times n$ matrix:

$$C = B^{-1}A \tag{1}$$

where B is the matrix of basis vectors and A is the matrix of coefficients augmented by the right-hand-side.

The term "revised" of the "revised simplex method" is used because the matrix multiplication, $B^{-1}A$, is not performed. Instead, the inverse B^{-1} is maintained explicitly, and is used to compute only those elements of C that are critical to

the simplex algorithm. Thus, if the top row of C represents the relative cost factors (δ_j's), we can compute them by applying the top row of B^{-1} to A. The minimum of these δ_j's determines the pivotal column of C which we compute by applying B^{-1} to the corresponding column in A. In a formal sense then, the original simplex method requires us to recompute an $m \times n$ matrix each iteration while the revised form requires us only to recompute one row and one column and to update the right-hand-side and the $m \times m$ inverse matrix B^{-1}.

This apparent advantage is sometimes less than real, but the revised form offers other benefits that aid greatly in the solution of large problems. If, for example, C is too large to be kept in the computer's high speed memory device, the revised form would have obvious advantages. When high speed memory is no longer adequate to hold B^{-1}, the revised form is in trouble. The product form of the inverse is a help in this case [14].

As noted, with the revised simplex method we do not update C; we only update B^{-1}; C given by (1) is never computed. With the revised simplex method, with the inverse in product form, we do not update B^{-1} either. Instead we compute a sequence of elementary matrices $\eta_1, \eta_2, \ldots$ where an additional η_i is computed on each "re-inversion" iteration and on each simplex iteration. After re-inversion and k simplex iterations B^{-1} is given by (2) below. It is never computed.

$$B^{-1} = \eta_{m+k}\, \eta_{m+k-1} \cdots \eta_{m+1} (\eta_m \eta_{m-1} \cdots \eta_1) \qquad (2)$$

When the price vector (i.e. the top row of B^{-1}) is to be computed, the product terms of (2) are premultiplied by $p = (1, 0, 0, \ldots, 0)$ and the multiplications executed by working from left to right so that each multiplication is that of a $1 \times m$ vector by an elementary $m \times m$ matrix. When the pivotal column of C is to be computed, the product terms of (2) are post-multiplied by the selected

column of A. The calculations are performed from right to left so that each multiplication is that of an elementary $m \times m$ matrix by a $m \times 1$ vector.

Most problems require so many iterations that the product form becomes impractical unless the product is periodically reduced to its fewest factors. Unless there happen to be unit basic columns, this minimum number is m as shown by (2). The usual practice is to "throw away" all the η's and to reconstruct m new ones from the current basis B. In LP parlance this process is called "re-inversion" or simply, "inversion".

If the usual simplex rules are adhered to and we neglect the occurrance of "ties", the η's outside the parentheses in (2) are determined uniquely. These are the η's produced during the simplex iterations and their values result from the simplex pivot criteria. The η's inside the parentheses in (2) are the ones produced during inversion where different criteria may be applied. The most successful criteria seem to be those that seek two major goals; in their purest form they are:

(a) Numerical accuracy

(b) Few non-zero elements

We will return to (a) later. In order to appreciate (b), it must be noted that most large LP matrices have a low percentage of non-zero elements. (A density of less than 5% is not unusual.) If the η's can be kept relatively sparse they will not only imply less arithmetic, but it will also be possible to hold them in less computer store by the use of a compact format containing only the non-zero's (in the relevant column) and their row indices.

As far as the authors know there is no known theoretical way (except perhaps exhaustive search of pivots) for finding the product form of the inverse that has the minimum number of non-zero elements. Some very simple procedures, however, in practice have shown quite remarkable improvements over those where no attention to (b) is given. Selecting the pivot columns of B in the order of increasing density often has a profound effect on the percentage of non-zeros in the inverse product form. (In practice, the entire matrix A may be pre-ordered by density which makes the ordering of any basis B automatic.)

One of the measures we have used in connection with criteria (b) shown above is the percent increase in non-zero elements in the representation of the inverse compared with the number of non-zero elements in the basis matrix before inversion.

M3 uses the product form of the inverse and early versions selected pivot columns during inversion by considering basis columns in the order they were originally given and the pivot row was selected by taking the largest pivot in absolute value on the remaining rows. On typical problems in the 2-5% density class, it was found that often there was an increase of about 5-10 fold in non-zero elements after inversion. Using just the simple technique of presenting the columns by density (actually only a partial sort on large problems) it was found that this increase in non-zeros could be cut by a factor of about two.

A related idea is the so-called "merit" sort which orders the columns by increasing merits. The total merit in a column is the sum of merits assigned to each non-zero element in the column. The merit of a non-zero element in row i is the number of non-zero elements in row i. Some other techniques of this kind

are described in Larsen [15] and Tewarson [16]. The compact inversion technique used in M5 was influenced by the above experience with various criteria. M5 uses the "Elimination Form of the Inverse", see Markowitz [17]. The notable thing about this evolution of computational improvements is that each one introduces something additional to be done implicitly rather than explicitly: the revised method does not explicitly compute C, the product form method does not explicitly compute B^{-1}, the Markowitz' method does not explicitly compute the elements of η_i in rows previously used for pivoting.

It has long been known that triangularization followed by back solution requires less calculation than full diagonalization in 100% dense systems. Our experience indicates that the same holds for sparse systems. Application of the triangularization approach to a staircase structured basis is found in [18]. In [19] Bartels discusses numerical accuracy of elimination procedures.

Elements on rows already used for pivoting are not eliminated in M5 but are stored as a matrix T which is <u>triangular</u> with respect to the pivot diagonal. The transformation matrices E_i are similar in structure to a product form transformation matrix with the exception that there are no elements above the diagonal, the diagonal being defined from left to right from successive pivot positions. Ignoring a permutation matrix which in practice is effected by maintaining the indices of pivot positions, the following relationships hold:

$$E_m E_{m-1} \cdots E_1 B = T$$

$$B^{-1} = T^{-1} E_m E_{m-1} \cdots E_2 E_1 \qquad (3)$$

As with the product form, after k simplex iterations, we will have:

$$B^{-1} = \eta_{m+k} \eta_{m+k-1} \cdots \eta_{m+1} (T^{-1} E_m E_{m-1} \cdots E_1) \qquad (4)$$

The device of course is not to compute T^{-1} but only T. In M5 the elements of the triangular matrix T are stored by columns. When the top row of B^{-1} is desired, we first compute from left to right:

$$P_1 = P\eta_{m+k}\eta_{m+k-1}\cdots\eta_{m+1}$$

then

$$P_2 = P_1T^{-1}$$

is found by "backsolving" the triangular system $P_2T = P_1$ and finally, we compute the product, again from left to right:

$$P_2E_mE_{m-1}\cdots E_1$$

The selected column of C is calculated in an analogous right to left manner. The matrices T and E_i are not unique so that there exists the opportunity to seek goals (a) and (b). With this in mind, we considered several pivot selection rules.

Pivot Selection

In our initial investigations five pivot selection algorithms were considered, which are described below. A comparison between them was made by performing symbolic triangularization on several bases taken from actual applied problems and keeping track of the position of non-zero elements. This symbolic triangularization was carried out on the computer. The proliferation of non-zero elements was all that was recorded. The arithmetic of the reduction procedure was not performed. One consequence of this is that any cancellation on non-zero elements giving rise to a zero in any particular spot was assumed not to occur.

The five pivot-selection algorithms considered have one characteristic in common. They are all dynamic in the sense that at each stage, the next pivot is chosen in the light of the updated matrix. In this way, the criteria differ from some of the techniques mentioned earlier. At each pivot-selection step, the choice depends only on properties of that part of the updated matrix made up of rows and columns not yet made use of for pivoting and, of course, the pivot choice is made from this submatrix. The number of non-zero elements in each such partial row and column is recorded and kept current at each pivot step.

The pivot-selection criteria considered are now described. Of all the positions in the as yet unpivoted portion of the updated matrix which contain a non-zero element:

1) Select as next pivot row, the row with the minimum number of non-zero entries. In the case of ties, take the first such row. As the pivot column, take the first column which contains a non-zero entry in this pivot row.

2) Select next pivot row as in 1). Of all columns which have a non-zero entry in the pivot row, select the first column which contains the minimum number of non-zero entries.

3) Consider the subset of rows {R} which have the minimum number of non-zero entries. There will be only one such row, unless ties occur. Of all the columns which have at least one non-zero entry on a row of {R}, select the first column which contains the minimum number of non-zero entries. This is the pivot column. The first row of {R} containing a non-zero entry in the pivot column is taken as the pivot row.

4) Select that element as pivot, such that the product of the number of other non-zero elements in its row, times the number of other non-zeros in its column, is a minimum. This criterion is due to H. Markowitz [17].

5) Take that position containing a non-zero entry as pivot which creates the fewest <u>additional</u> non-zero entries, when carrying out the reduction step.

6) to 10). Apply the same rules as above to the transposed matrix. Note that 9) could lead to a different choice of pivot than 4) because of the tie-breaking rules; similarly 10) differs from 5)

In each case, the total number of non-zero entries in the E_i and T matrices is compared with the number of non-zeros in the original matrix B of basis vectors.

The results for two matrices are shown in Table 1. They are typical of results that we have observed on other matrices which are not presented here because of incomplete information regarding the source of the matrices or incomplete analysis on our part. In our opinion, it would be worthwhile for some one to collect a wide variety of matrices used in LP applications (not randomly generated ones) to verify whether or not these results are typical.

	Matrix I		Matrix II	
SIZE	62 × 62		216 × 216	
NUMBER OF NON-ZERO ENTRIES	348		1400	
DENSITY	9%		3%	
ALGORITHMS	NON-ZERO ENTRIES IN E_i AND T	% INCREASE	NON-ZERO ENTRIES IN E_i AND T	% INCREASE
1) Minimum Row - Column with 1st non zero entry.	378	8.6	1456	4.0
2) Minimum Row Minimum Column	364	4.6	1485	6.1
3) Minimum Rows Minimum Column	366	5.2	1424	1.7
4) Minimum Product [Markowitz]	378	8.6	1492	6.6
5) Minimum Increase	420	20.7	1553	10.9
6) Same as 1 on transpose	408	17.2	1654	18.1
7) Same as 2 on transpose	361	3.7	1493	6.6
8) Same as 3 on transpose	361	3.7	1461	4.4
9) Same as 4 on transpose	381	9.5	1514	8.1
10) Same as 5 on transpose	366	5.2	1523	8.8

TABLE 1. PIVOT SELECTION CRITERIA

Our intuition would have probably led us to algorithm 5), which did not show up very favorably in our admittedly few experiments. We chose to adopt algorithm 8) for the LSP code M5.

It is interesting to note that our experiments indicated the above technique performs much less favorably on sparse matrices whose non-zeros have been generated in a random manner. Symbolic matrices of 5% density and of orders 50, 200 and 400 were generated using the pseudo random number generator based on

$$X_{i+1} = aX_i \pmod{2^{35}} \quad \text{where} \quad a = X_o = 5^{13}$$

The location of a non-zero element in the matrix was determined by randomly selecting its row and its column. To insure non-singularity, symbolic non-zero elements were first located along the main diagonal.

The results using algorithm 8) are as follows:

	Random Matrix 1	Random Matrix 2	Random Matrix 3	Matrix I	Matrix II
SIZE	50	200	400	62	216
DENSITY	5%	5%	5%	9%	3%
INCREASE OF DENSITY AFTER TRIANGULARIZATION USING ALGORITHM 8)	13%	333%	480%	3.7%	4.4%

These results are evidence that general conclusions should not be drawn from linear programming algorithm experiments which have been performed using matrices whose elements have been generated in a random manner.

The exact form of the algorithm implemented is given below. For numerical accuracy, the above procedure is modified if the pivot chosen is less in absolute value than a specified tolerance value (see step 3). The test and procedure in the case of an ill-conditioned basis is omitted.

STEPS IN INVERSION PROCEDURE

1 Count and store number of non-zeros in basis matrix by column and row.

2 Restrict the candidates for pivot to non-zero elements in columns and rows not previously used for pivot whose non-zero column-counts are minimal.

Further restrict the candidates to those whose absolute values are $\geq$ a given tolerance value; if none then those whose absolute values are maximal.

Further restrict the candidates to those being in rows whose non-zero row-counts are minimal.

Among the latter select as pivot the one with the lowest column-index and then the lowest row-index.

3 Generate transformation column E_i and corresponding column of T.

4 Update residual matrix, adjusting the non-zero row and column counts of elements in non-pivoted rows and columns.

5 Repeat steps 2 through 5 until all columns have been used for pivoting.

The implementation ofthis algorithm also presented us with some interesting systems problems in data handling. We choce to restrict the application of the code to problems where inversions could be carried out entirely in the high

speed storage of the computer. As noted earlier both M3 and M5 are used for problems up to about 400 rows. With the density and size problems encountered M5 has carried the elimination form of inverse successfully in core.

Some comparison times of M3 vs M5 are given in Table 2.

TABLE 2. COMPARISON OF RESULTS [M3 vs M5]

PROBLEM SIZE			Run Time M3	Run Time M5	M5 as % of M3
99	165	Complete Problem	67 secs	42 secs	63%
166	744	Average Inversion	21 secs	10 secs	48%
		Average Iteration	3.1 secs	1.0 secs	32%
264	1118	Average Inversion	100 secs	25 secs	25%
		Average Iteration	4.7 secs	1.2 secs	26%

REFERENCES

1. The M3 Linear/Separable Programming Code Manual, Unpublished Standard Oil Company of California Report.

2. The M5 Linear/Separable Programming Code Manual, Unpublished Standard Oil Company of California Report.

3. DANTZIG, G.B., Notes on Linear Programming: Parts VIII, IX, X - Upper Bounds, Secondary Constraints, and Block Triangularity in Linear Programming, The RAND Corporation, Research Memorandum RM-1367, October 4th, 1954. Published in Econometrica, Vol. 23, No. 2, April 1955, p. 174-183.

4. GASS, S.I. and T.L. Saaty, "The Computational Algorithm for the Parametric Objective Function", Naval Research Logistics Quarterly, Vol. 2, No. 1, June 1955.

5. ORCHARD-HAYS, W., SCROL, A Comprehensive Operating System for Linear Programming on the IBM 704, Users Reference Manual, CEIR.

6. WOLFE, Philip, "The Composite Simplex Algorithm: Notes on Linear Programming and Extensions - Part 64", The RAND Corporation, Research Memorandum RM-3579-PR April, 1963. Also in SIAM Review, Vol. 7, No. 1, January 1965.

7. McKNIGHT, R.D. and R.P. Harvey, "The Upper Bounding Algorithm for Separable Programming: Notes on Separable Programming Series", Standard Oil Company of California [in preparation].

8. MILLER, C.E., "The Simplex Method for Local Separable Programming", Standard Oil Company of California, August 1960. Also published in [R. Graves and P. Wolfe eds.] Recent Advances in Mathematical Programming, McGraw-Hill, 1963.

9. McKNIGHT, R.D., "The Representation of the Product of Certain Variables without Separation: Notes on Separable Programming Series", Standard Oil Company of California, May 9th, 1962.

10. McKNIGHT, R.D., "Polynomial Column Generation with an Application to Quadratic Programming by Successive Approximation: Notes on Separable Programming Series", Standard Oil Company of California, May 21st, 1962.

11. McKNIGHT, R.D., "Nonconvexities Due to Separation With an Example from Probabilistic Programming: Notes on Separable Programming Series", Standard Oil Company of California, November, 1964.

12. HARVEY, R.P. and R.D. McKnight, "An Algorithm for the Automatic Regulation of Tolerance Values in Linear Programming Codes", Standard Oil Company of California Memorandum [in preparation].

13. DANTZIG, G.B., "Linear Programming and Extensions", Chapter 9, Princeton University Press, 1963.

14. DANTZIG, G.B. and W. Orchard-Hays, "Alternate Algorithm for the Revised Simplex Method Using Product Form for the Inverse", The RAND Corporation, Research Memorandum, RM-1268, November 19th, 1953.

15. LARSEN, L.J., "A Modified Inversion Procedure for Product Form of the Inverse Linear Programming Codes", Comm. of the ACM, Vol. 5, No. 7, July 1962.

16. TEWARSON, R.P., "On the Product Form of Inverses of Sparse Matrices", SIAM Review Vol. 8, No. 3, July 1966.

17. MARKOWITZ, H.M., "The Elimination Form of the Inverse and Its Application to Linear Programming", The RAND Corporation, Research Memorandum, RM-1452, April 8th, 1955. Also published in Management Science, Vol. 3, No. 3, April 1957, p. 255-269.

18. DANTZIG, G.B., "Compact Basis Triangularization for the Simplex Method", Operations Research Center, University of California, Berkeley, RR-33 (August 1962); also in (R.L. Graves and P. Wolfe, eds.) Recent Advances in Mathematical Programming, McGraw-Hill, New York 1963.

19. BARTELS, R.H., "A Numerical Investigation of the Simplex Method", Computer Science Department, Stanford University, Technical Report No. CS-104, July 31, 1968.

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Department of Operations Research Stanford University STANFORD, California 94305	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

Sparse Matrix Techniques in Two Mathematical Programming Codes

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Technical Report

5. AUTHOR(S) *(Last name, first name, initial)*

Dantzig, George B., HARVEY, Roy P., McKNIGHT, Robert D., SMITH, Stanley, S.

6. REPORT DATE January 1969	7a. TOTAL NO. OF PAGES 15	7b. NO. OF REFS 19
8a. CONTRACT OR GRANT NO. N-00014-67-A-0112-0011 b. PROJECT NO. NR-047-064 c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) Technical Report No. 69-1 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*	

10. AVAILABILITY/LIMITATION NOTICES

Distribution of this document is unlimited

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Logistics and Mathematical Statistics Branch, Mathematical Sciences Division, Office of Naval Research, WASHINGTON, D.C. 20306

13. ABSTRACT

The authors empirically compared ten pivot selection rules for representing the inverse of a sparse basis in triangularized product form. On examples drawn from actual applications, one of the rules yield inverses that were only slightly less sparse than the original basis. This rule was used in the M5 mathematical programming system and has resulted in substantial reduction in running times.

DD FORM 1 JAN 64 1473

Unclassified
Security Classification

Unclassified
Security Classification

KEY WORDS	LINK A ROLE	LINK A WT	LINK B ROLE	LINK B WT	LINK C ROLE	LINK C WT
Sparse Matrix						
Mathematical Programming						
Linear Programming						
M5 Mathematical Programming System						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through _____."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through _____."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through _____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS)*, *(S)*, *(C)*, or *(U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, roles, and weights is optional.

DD FORM 1 JAN 64 1473 (BACK)

Unclassified
Security Classification