PROGRAM ABSTRACTS/ALGORITHMS

MOP: Matrix operations program

ALAN B. ZONDERMAN

University of California, Berkeley, California 94720

MOP performs matrix algebraic operations. All MOP instructions are input in a free-field fashion. Input consists of a regular form of algebraic syntax for matrix addition, subtraction, and multiplication. Various mathematical and utility procedures are available for operations such as inversion, eigen analysis, reading, writing, and printing. Matrices are referenced by alphabetic names with no embedded blanks. No distinction is made between matrices and vectors. Column or row vectors are easily accommodated as matrices with only one column or row, respectively.

In addition, the user may set row, column, or dimension specifications to nonnegative integers. These variables are referenced by alphanumeric names that are separate from matrix names. Integer variables may replace size specifications in most MOP procedure calls.

There are two input and two output files. All MOP commands are read from INPUT and all MOP output is placed on OUTPUT. Matrices may be read from INPUT or ALTINP using the READ procedure; matrices may be written to OUTPUT or ALTOUT using the WRITE procedure.

MOP instruction lines may be no more than 80 characters in length. No continuation lines are allowed.

Algebraic Expressions. Matrix addition, subtraction, and multiplication follow customary algebraic syntax. The only restriction to forming such expressions is that the result must be on the left. Conformity of operands is checked. It is a fatal error when the operands in an algebraic statement do not conform for the intended matrix operation. The possible operations are $A = B + C$, $A = B - C$, $A = B * C$, and $A = B$, where A , B , and C are matrices and where they conform for the indicated operation. In each of these, the operands must exist prior to the operation. The result of the operation is created if it does not exist. If the result exists, it is replaced (overwritten) if it is the appropriate size;

I thank William Meredith for his help in suggesting tests for the correctness and accuracy of this program. His constant pursuit of additional options led to many of the procedures in MOP. In addition, I thank his graduate students Roger Milsap and John Tisak for their attempts to make MOP fail and for their tests against other programs. Support was provided by NIH Postdoctoral Fellowship NICHD Grant HD-05351 under William Meredith's sponsorship at the University of California at Berkeley. Reprints and information are available from Alan B. Zonderman, Department of Psychology, Johns Hopkins University, Baltimore, Maryland 21218.

otherwise, it is deleted and recreated. The user need not take note of this except when the result must be saved for later processing. The left-hand side may also be an operand for addition and subtraction. However, it is a fatal error when the result is also an operand in a matrix multiplication.

Integer Variable System. The facility for writing general-application MOP routines is provided by setting MOP integer variables to take the place of absolute numbers in various procedure calls. MOP integer variables are separate from matrix names and storage. Therefore, matrix names and integer variable names do not conflict; there may exist identically named matrices and variables without confusion. In general, procedures that require as parameters row, column, or size dimensions accept MOP integer variables in place of an absolute integer specification. Exceptions to this rule are noted in the detailed description of MOP procedures.

As a consequence of the separation of matrix names and integer variable names, it is not possible to use the latter in computations. Scalars are formed either (1) by direct input, as the result of a transfer from integer storage, or (2) through a computation that results in the formation of a scalar. The distinction between the two forms of storage is that matrices are used in numeric computation, while integer variables are used as parameters in procedure calls.

The details of specifying MOP integer variable values are described in the MOP user's manual. There is a limit of 100 MOP integer variables.

Procedures. A variety of mathematical and utility procedures are available. Procedure calls follow the regular syntax,

NAME(P1, P2, . . . , Pn),

where NAME is the name of the procedure to be executed, and P1, P2, . . . , Pn are the arguments of the procedure. Several procedures require additional input (e.g., a format). When such input is required, it follows immediately after the procedure call line. Several procedures operate on Gramian matrices. Gramian is defined as a positive semi-definite matrix or, alternatively, a matrix with no negative eigen values. An additional restriction placed on the definition of Gramian by this program is that the matrix is symmetric. Examples of MOP procedures organized by functions are given in Table 1.

Dynamic Memory Management. MOP adjusts central memory as it is required for the storage of matrices and results. The user should set the field length to load to the minimum for execution (approximately 30,000 octal

Table 1
Examples of MOP Procedures Organized by Functions

Matrix Computations	
BILINEAR	Evaluate a bilinear form.
DIAGPOST	Postmultiply a matrix by a diagonal matrix.
DIAGPRE	Premultiply a matrix by a diagonal matrix.
EIGEN	Compute eigen values and eigen vectors.
INVERT	Invert a square matrix.
INVERTSYM	Invert a Gramian matrix.
KRONECKER	Compute a Kronecker product.
PREPSTDIA	Pre- and postmultiply a matrix by a diagonal matrix represented in a vector.
TRANSPPOST	Postmultiply a matrix by the transpose of a matrix.
TRANSPRE	Premultiply a matrix by the transpose of a matrix.
TRIANGLE	Triangular factorization of a Gramian matrix.
Matrix Formation	
FORM	Form a matrix by specifying its on and off diagonal elements.
IDENTITY	Form an identity matrix.
UNIT	Form a unit vector.
ZERO	Form a null matrix.
Matrix Manipulation	
DIAGEXT	Extract the diagonal of a square matrix.
DIAGINS	Insert a vector into the diagonal of a matrix.
SUBMATRIX	Submatrix removal or insertion.
Output Listing Control	
COMMENT	Place a comment in the output listing.
PRINT	Print a matrix.
Input-Output	
READ	Input a matrix.
WRITE	Output a matrix.
Integer Variables	
SET	Create, compute, delete, or print the value of an integer variable.
TRANSFER	Transfer a MOP integer variable to a (1 by 1) matrix.
MOP Control	
DELETE	Remove a matrix from MOP storage.
RENAME	Rename a matrix.

words). Prior to the execution of the first input instruction, the field length is expanded to accommodate 5,000 words of matrix storage. Subsequently, additional memory is requested from the operating system in minimum 1,000-word segments.

Installation Notes. MOP is written in FORTRAN and COMPASS for the FTN 4.0 compiler of CDC 6000, 7000, and CYBER series computers. Because several not

inconsequential routines are coded in assembly language, the program will not produce a correct object file under any other circumstances. There are no plans at present to rewrite these sections for portability. In addition, the FTN (not RUN) FORTRAN compiler must be used because the COMPASS sections of MOP are incompatible with RUN. Experience on a CDC 6600 and a CDC 6400 has shown that the OPT=2 compiler option produces a somewhat faster and somewhat smaller absolute binary than the default optimization option.

No attempt is made in this version to intercede system messages. If the user creates a situation in which, for example, the square root of a negative number is calculated, the FORTRAN error system and not MOP issues a warning. In particular, an incorrectly specified format statement or a file mark encountered before one is expected will result in the appropriate system error message. This may create some confusion among users who are unaccustomed to such error messages. A brief consultation with either the FTN manual or the local computer center staff may aid in the interpretation of such messages.

Accuracy. A check on the internal accuracy of several MOP procedures was performed on the 24-variable Holzinger-Swineford correlation matrix. The inversion routines were examined by multiplying the correlation matrix by its inverse and subtracting from an identity matrix. The resulting difference was zero to 12 decimal places. In similar fashion, the triangular factoring routine was tested by subtracting from the correlation matrix the product of the lower and upper triangular matrices. The result of this manipulation was a matrix, all elements of which were zero to 12 decimal places. In addition, the determinants calculated by the inversion and triangular factoring routines were identical. Finally, the eigen analysis procedure was checked by subtracting from an identity matrix the difference between the product of the correlation matrix and the matrix of eigen vectors, and the matrix of eigen vectors postmultiplied by a diagonal matrix of eigen values. The result of this computation was a matrix of zeros to 12 decimal places. It is apparent that these routines are numerically accurate, approaching the limit of precision of a single word of storage.

Availability. MOP source and test decks and a complete user's manual are available from the author at the cost of reproduction and mailing.

(Accepted for publication June 10, 1979.)