Programmatuur-sectie

C O M M A

Computations on more matrices

Frank B. Brokken (University of Groningen)

Introduction

The program COMMA is a computer program which may be used to solve problems related to optimizing factorial invariance between two or more matrices. COMMA is one of the products of the project "Invariantie onderzoek" and it is closely related to the dissertation by Ten Berge (1977a). Ten Berge's terminology will be used in this article for the descriptions of the different rotation procedures of COMMA.

In COMMA factorial invariance may be assessed by means of the coefficient of congruence (phi) (Tucker, 1951):

$$\text{phi}(\underline{a},\underline{b}) = \underline{a}'\underline{b}\,(\underline{a}'\underline{a}\underline{b}'\underline{b})^{-\frac{1}{2}} \qquad (1)$$

where $\underline{a}$ and $\underline{b}$ are n x 1 vectors containing for instance the loadings of the same n variables on two factors.

According to Ten Berge (1977a) this coefficient is to be preferred as an index for assessing factorial invariance to other indices when only the factor loadings are available. However, when factor scores of the same subjects are available, one should compute the Pearson correlation between the factor scores on the two factors. Fortunately, when the vectors $\underline{a}$ and $\underline{b}$ in formula (1) have zero means, phi reduces to the Pearson correlation.

Available Rotation Procedures

Although COMMA was designed to optimize factorial invariance for more than two matrices, it is possible to perform several types of analyses on two matrices. Using COMMA, the following analyses may be performed on two matrices:

Problem 1. Simultaneous one-sided orthogonal inproduct rotation.

Here the problem is to find an orthonormal k x k matrix T which maximizes

$$g(T) = \mathrm{tr.}(A_1'A_2T), \tag{2}$$

where A_1 and A_2 are n x k matrices (e.g., factor loadings). In the sequel the matrices A_1 and A_2 will be used without further denotation, they always refer to n x k datamatrices.

Problem 2. Simultaneous one-sided orthogonal Procrustes rotation.

Here the problem is to find an orthonormal k x k matrix T which minimizes

$$g(T) = \mathrm{tr.}(A_1 - A_2T)'(A_1 - A_2T). \tag{3}$$

The solution of this problem is identical to the solution of problem 1.

Problem 3. Simultaneous two-sided orthogonal inproduct rotation.

Here the problem is to find two orthonormal k x k matrices T_1 and T_2 maximizing

$$g(T_1,T_2) = \mathrm{tr.}(T_1'A_1'A_2T_2). \tag{4}$$

It is noted that COMMA does not yield the so-called "natural solution" of this problem. Instead, both matrices A_1T_1 and A_2T_2 are rotated by a common rotation matrix W which leaves the function g unchanged but which will rotate the matrix $A^* = \begin{bmatrix} A_1T_1 \\ A_2T_2 \end{bmatrix}$ to a varimax position.

Problem 4. Simultaneous two-sided orthogonal Procrustes rotation.

Here the problem is to find two orthonormal k x k matrices T_1 and T_2 minimizing

(5) $$g(T_1,T_2) = \text{tr.}(A_1T_1 - A_2T_2)'(A_1T_1 - A_2T_2).$$

The solution of this problem is identical to the solution of problem 3.

Problem 5. Simultaneous two-sided orthogonal congruence rotation after orthonormalization.

After transforming A_1 to X_1 by $X_1 = A_1(A_1'A_1)^{-\frac{1}{2}}$ and transforming A_2 to X_2 by $X_2 = A_2(A_2'A_2)^{-\frac{1}{2}}$, X_1 and X_2 may be rotated by the procedures used in solving problem 1 or problem 3. However, the "natural solution" of the solution of the two-sided rotation problem 3 cannot be obtained using COMMA. Instead, X_1T_1 and X_2T_2 will be varimax-rotated by a common matrix W (cf. problem 3).

The problems mentioned thus far may be used in cases where there are two matrices. Often, however, the researcher has more than two datamatrices (e.g., the same test is given to three groups of children). In these cases COMMA may be used to assess the invariance of the datamatrices (e.g., factor loadings) all at once. Ten Berge (1977a, p. 39) gives three functions to be used in assessing factorial invariance when there are more than two matrices:

a) The generalized Procrustes function

(6) $$f(T_1,T_2,\ldots T_m) = \sum_{i<j} \text{tr.}(A_iT_i - A_jT_j)'(A_iT_i - A_jT_j),$$

b) The generalized inproduct function

(7) $$f(T_1,T_2,\ldots T_m) = \sum_{i<j} \text{tr.}(T_i'A_i'A_jT_j),$$

c) The generalized congruence function

(8) $$f(T_1,T_2,\ldots T_m) = \sum_{i<j} \sum_{p=1}^{k} \text{phi}(A_it_{ip},A_jt_{jp})$$

Depending on the rotation problem, function (6) will be minimized while the functions (7) and (8) will be maximized. COMMA may be used in finding solutions for the following problems when there are more than two matrices:

Problem 6. Generalized simultaneous orthogonal inproduct rotation.

Here the problem is to find orthonormal matrices $T_1, \ldots T_m$ maximizing (7). The mathematical solution of this problem is given by Ten Berge (1977a) and is not repeated here. It should be noted that the solution is unique up to a common rotation W. One possibility is to determine W as the varimax rotation matrix for the super matrix

$$A^* = \begin{bmatrix} A_1T_1 \\ \hdashline A_2T_2 \\ \hdashline \vdots \\ \hdashline A_mT_m \end{bmatrix}.$$

Another solution (W = I) may be called the "natural solution". Both the varimax solution and the natural solution may be obtained using COMMA.

Problem 7. Generalized simultaneous orthogonal Procrustes rotation.

As in the case where there are two matrices (problem 4), this problem reduces to the orthogonal inproduct rotation problem. The function (6) has to be minimized.

Problem 8. Generalized simultaneous orthogonal congruence rotation after orthonormalization.

After replacing the matrices A_i by $X_i = A_i(A_i'A_i)^{-\frac{1}{2}}$ (i = 1, ... m) this problem is reduced to problem (6) on the X-matrices. The matrices X_i may be rotated by a common rotation matrix W, which will leave the sum of phi-coefficients unaffected.

This solution represents the best currently available generalization of the canonical correlation problem, formulated as a simultaneous rotation problem. Especially when the canonical variates are to be rotated to "simple structure" (e.g., by means of the varimax criterion), this is a useful approach to the generalized canonical correlation problem (cf. Cliff & Crus, 1976).

Approximate solutions

In addition to the proper and exact solutions of the rotation problems presented above, several other problems may be solved by using an approximate solution. Instead of presenting these

problems and their solutions in extenso, they will only be mentioned and a reference to Ten Berge (1977a) will be given.

The following problems may be solved approximately:

a) Two matrices.

1) Simultaneous two-sided orthogonal inproduct rotation after orthonormalization, followed by a common varimax rotation (Ten Berge, 1977a, p. 19).

2) Simultaneous two-sided orthogonal Procrustes rotation after orthonormalization, followed by a common varimax rotation (Ten Berge, 1977a, p. 19).

b) Three or more matrices.

1) Generalized simultaneous orthogonal inproduct rotation after orthonormalization (Ten Berge, 1977a, p. 41).

2) Generalized simultaneous orthogonal Procrustes rotation after orthonormalization (Ten Berge, 1977a, p. 41).

Upper bounds to the obtained invariance.

In the case of more than two matrices, the solutions of the problems satisfy only a necessary condition for the maximum (minimum) of the functions to be maximized (minimized). However, Ten Berge (1977b) gives two upper bounds to the functions which are both computed by COMMA. Comparison of these upper bounds and the obtained "criterion value" will thus give an indication of the success of the rotation procedure. If the criterion value is close (or even equal) to the lowest upper bound, then the absolute maximum (minimum) of the function to be maximized (minimized) is (almost) reached and vice versa. The proofs of the two upper bounds may be found in Ten Berge (1977b).

The first upper bound.

Let $A_i'A_j = P_{ij}D_{ij}Q_{ij}'$ be an Eckhart-Young decomposition of $A_i'A_j$, then the first upper bound to the inproduct function (7) is given by $\sum_i \sum_j tr.D_{ij}$. So if f denotes the value of the function (7) then the following relation will always hold true:

$$f \leq \sum_i \sum_j tr.D_{ij} \qquad (9)$$

The second upper bound.

Let $\overline{A'A}$ denote the km x km super matrix

$$\overline{A'A} = \begin{vmatrix} 0 & A_1'A_2 & \cdots & A_1'A_m \\ A_2'A_1 & 0 & \cdots & A_2'A_m \\ \vdots & \vdots & \ddots & \vdots \\ A_m'A_1 & A_m'A_2 & \cdots & 0 \end{vmatrix} , \tag{10}$$

let $\overline{A'A} = PDP'$ be an eigenvector-eigenvalue decomposition of $\overline{A'A}$ ($d_i \leq d_j$ for $i < j$), then the second upper bound is given by

$$f \leq \frac{m}{2} \sum_{i=1}^{k} d_i \tag{11}$$

Data and Syntax Diagrams

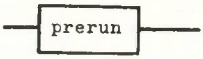
The program COMMA was written in the language Algol-60. Consequently, numerical data for COMMA should be given according to the Algol syntax for (real) numbers. Generally this will not present any difficulty as the syntax is very "natural", i.e., leading zero's may be omitted, numbers may contain a minus sign etc.. Also, the positions of the numbers on punching cards is not fixed, but the numbers must be separated by commas, semicolons or by at least two blank spaces. However, one special Algol number symbol deserves some attention. The quotation mark (") may appear in a number meaning "(times) ten to the power".E.g., 3"-4 denotes the number 0.0003 and "-4 denotes the number 0.0001.

Beside the Algol-syntax for real numbers, the input for the program should be structured according to the "COMMA-syntax". In-

put structured according to this syntax will be accepted by the program and will result in a proper run of COMMA, while all other input will result in an early abort of the program. In these cases a syntactical error is said to have occurred for which COMMA will frequently issue an errormessage.

The COMMA-syntax is given by so called "syntax diagrams", containing terminal and non-terminal symbols. By successively replacing the non-terminal symbols by their definitions, which will usually contain one or more terminal symbols, the non-terminal symbols disappear and correct COMMA input is generated.

In the syntax diagrams non-terminal symbols are represented by words written in lower-case letters and placed in rectangles, like

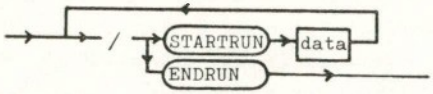
(12) 

On the other hand, the terminal symbols are set up in capital letters and are placed in rectangles having rounded angles, like

(13) 

Furthermore, all numbers, all literal symbols (e.g., the comma, the blank space (denoted by ^)) and the symbol / (denoting "continue on a new card or line of text") are terminal symbols.

The distinct elements of a syntax diagram are joined by lines containing small arrows denoting the seriation of the input elements. For example, in (14) the general form of the COMMA-in-

(14) 

put is depicted in the "COMMA-input" syntax diagram. The input may consist of one or more sequences of the word STARTRUN (placed on a new card or line of text), followed by data as described by the "data" syntax diagram, but the last card of the input must contain, on a new line of text or on a new card, the word ENDRUN.

The COMMA syntax

This section contains all syntax diagrams necessary for the generation of correct COMMA-input. Numbers in the syntax diagrams are represented by letters or abbreviations, set up in lower case letters and placed in rhombs. The meaning of these numbers will always be explained in the text.

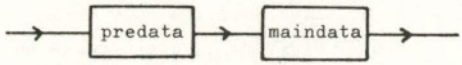
COMMA-input.

Diagram (14) shows the general form of the COMMA-input. This diagram was explained in the text immediately following (14) and is not repeated here.

data.

The non-terminal symbol "data" in (14) contains two non-terminal elements. The data syntax diagram is given in (15), showing

(15)

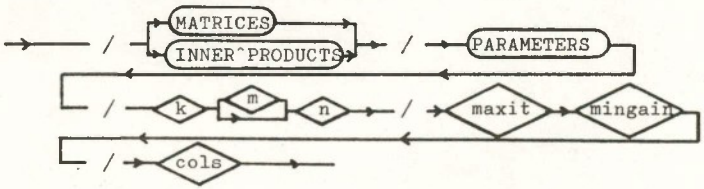


that all data for one problem are separated in two parts: predata and maindata.

predata.

The predata syntax diagram is given in (16). Starting on a

(16)



new card or line of text either the word MATRICES appears (i.e., the user supplies the program with the separate A_i matrices) or the word INNER PRODUCTS appears (i.e., the user intends to enter the $A_i^t A_j$ matrices in some form). The next card contains the word PARAMETERS, which parameters are given on the following three cards. These parameters are:

k: the number of matrices to be entered;

m: number of rows of the A_i matrices (only if MATRICES was given);

n: number of columns of the (inner product) matrices;

maxit: maximum number of iterations during the rotation of the matrices. One iteration corresponds to one computation of the function (7) in which, depending on the problem, the A_i matrices may represent transformed (e.g., orthonormal) matrices or the original matrices;

mingain: the minimal gain in the function (7) between two successive iterations necessary to start a new iteration.

Maxit and mingain together determine whether a new iteration cycle will be started. A new iteration is performed whenver the current iteration number is less than maxit and the gain of the function (7) between the last and the last but one iteration exceeds mingain. Maxit and mingain must always be specified, even if no rotation will be requested (cf. the NOGO option of the "maindata" syntax diagram). However, in these cases the actual values of these parameters are immaterial.

cols: the number of columns to be used in reading subsequent input information. E.g., 72: information beyond column 72 is to be ignored).

Example.

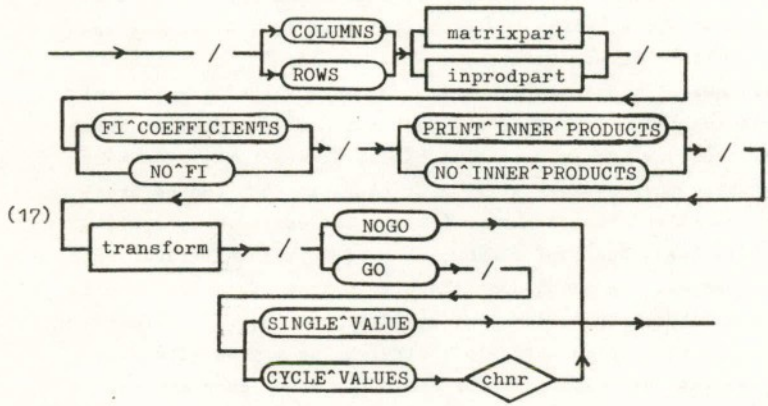
Using the syntax diagrams presented above, the following input may be constructed:

```
STARTRUN
MATRICES
PARAMETERS
3, 50, 10
15, "-4
80
```

From this input fragment COMMA will expect three matrices having 50 rows and 10 columns each. If the remaining input will request a rotation of these matrices, then the function (7) may be computed at most 15 times while each new value of (7) must be at least 0.0001 higher than the previous value of (7). The remaining input will be read over at most 80 columns.

maindata.

The data (elements of the A_i or $A_i'A_j$ matrices) may be entered in two major ways: after ROWS the elements of each matrix are to be entered in the order a_{11} to a_{1n}, a_{21} to a_{2n}, ... a_{n1} to a_{nn}, after COLUMNS the elements are to be entered in the order a_{11} to a_{n1}, a_{12} to a_{n2}, ... a_{1n} to a_{nn}. From the maindata syntax diagram (17) it can be seen that the program after the rows or columns specification chooses between the non-terminal "matrixpart" or



the non-terminal "inprodpart". It can do so thanks to the specification on the first card (terminal symbol) of predata.

Next the user may request the computation of the phi-coefficients between all columns of the data matrices and the printing of the inner product matrices may be requested. These parameters appear immediately after the non-terminals "matrixpart" and "inprodpart" thus indicating that they will become effective immediately after the input of the datamatrices. However, they will remain active during the complete run. Wherever phi-coefficients and/or inner products may be computed they will also be printed if requested so at this point.

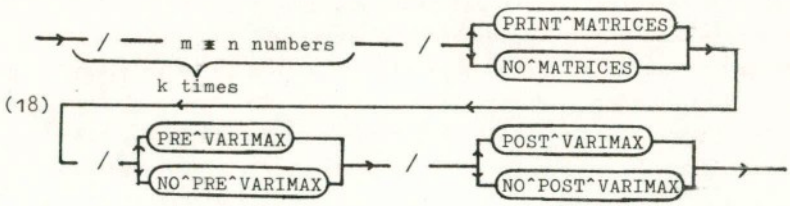
The datamatrices may be transformed in many ways (e.g., the matrices A_i may be orthonormalized by means of the transformation $X_i = A_i(A_i'A_i)^{-\frac{1}{2}}$). All possible transformations are described in the "transform" section.

The user who has nothing else in mind but to produce a nice

output showing his data (perhaps after performing some transformations) may leave the maindata section by presenting the word NOGO. On the other hand, the presentation of the terminal GO will activate the rotation procedures by which the current datamatrices will be transformed in such a way that the invariance between the individual matrices is maximized (i.e., function (7) or (8) will be maximized or function (6) will be minimized). During this rotation process the function (7) will be computed repeatedly. Presentation of the SINGLE VALUE card will result in merely printing the final value of (7), after convergence or after "maxit" (cf. the predata section) computations of (7). Presenting the card containing CYCLE VALUES results in the printing of the value of (7) each time it is computed, which happens once during each rotation cycle. However, the user must decide where these values will be printed. To this end he presents a so-called channel number "chnr". The easiest way is to substitute the value 61 for chnr, whereafter the individual function values will be printed on the line-printer, generally halfway through the output, immediately after the initial presentation of the data. Alternatively another value may be given. As this requires the preparation of another channel card and usually also the preparation of some more job control cards, both beyond the scope of this article, it was decided not to discuss this alternative within the context of this article.

matrixpart

If the terminal symbol MATRICES was given, the program will expect k datamatrices, having m rows and n columns. Syntax diagram (18) shows that each matrix has to start on a new line,



while the elements of the individual matrices must be given according to the ROWS or COLUMNS definition in maindata.

After the input of the k matrices the user either explicitly requests the printing of the datamatrices (by using PRINT MATRICES) or this printing will be supressed (NO MATRICES). This latter option is especially useful when the matrices have many elements.

As the datamatrices are available, another type of rotation is possible apart from the rotations performed to achieve the invariance of the matrices. The matrices may be varimax rotated, either separately before the invariance producing rotations or following the invariance rotations, in which case the super matrix (19) will be varimax rotated.

$$(19) \qquad A^{*} = \begin{bmatrix} A_1T_1 \\ \hline A_2T_2 \\ \hline \vdots \\ A_kT_k \end{bmatrix}$$

By separately varimax rotating the k matrices before the (iterative) invariance rotations the matrices are <u>generally</u> rotated to a position which speeds up this iterative process. On the other hand, the user may of course be merely interested in the matrices in their varimax positions.

These PRE VARIMAX rotations of the matrices produce a new set of datamatrices. During subsequent rotations or transformations of the matrices the varimax rotated matrices will be used and so subsequent rotation matrices must be applied to the matrices in their varimax positions.

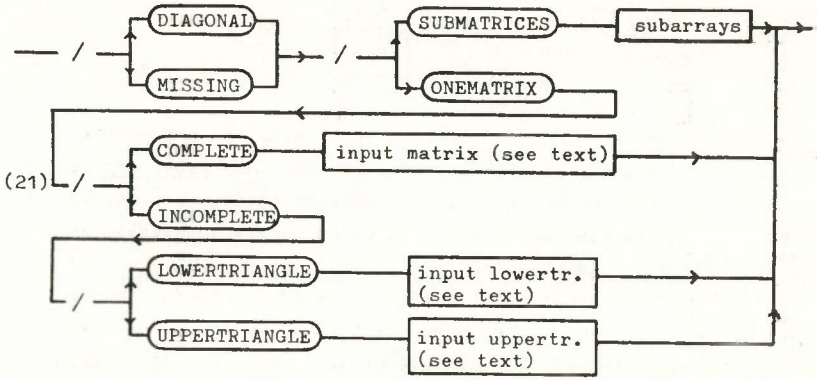
The varimax rotation of the matrix A^{*} (19) (POST VARIMAX) produces one rotation matrix for all datamatrices. The required rotation matrix W will be printed in addition to the k resulting rotation matrices obtained after the multiplication of the k previous rotation matrices by W.

<u>inprodpart</u>

In some cases the data matrices are not available while their inner product matrices are available. When there are k matrices, having n columns each, a kn x kn super matrix as depicted in (20) represents all inner product matrices. In diagram (20) i'j represents the n x n inner product matrix of data matrix i (transpos-

$$
(20)\qquad
\begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ k \end{array}
\begin{array}{c}
\begin{array}{cccc} 1 & 2 & \cdots & k \end{array} \\
\begin{bmatrix}
1'1 & 1'2 & \cdots & 1'k \\
2'1 & 2'2 & \cdots & 2'k \\
\vdots & \vdots & \vdots & \vdots \\
k'1 & k'2 & \cdots & k'k
\end{bmatrix}
\end{array}
$$

ed) and data matrix j. (Parts of) the inner product super matrix (20) may be entered according to the inprodpart syntax diagram (21). The order in which the individual elements of the inner product matrices of (20) will be entered has already been specified by the ROWS or COLUMNS terminal symbol of the maindata section. The order in which the individual inner product matrices will be entered is specified by using the inprod syntax diagram.



The user may:

a) Either omit the main diagonal inner product matrices (i.e., 1'1, 2'2, ..., k'k) by specifying MISSING or enter these matrices by specifying DIAGONAL.

b) Either enter the inner product matrices separately, one by one, by specifying SUBMATRICES, followed by the non-terminal "submatrices", or enter the supermatrix (20) as one matrix by specifying ONEMATRIX.

If ONEMATRIX is given and the user has available the complete supermatrix, then COMPLETE must be given followed by the input of the complete matrix (20) ($(kn)^2$ numbers). Frequently, however, the super matrix is not complete, but, as the super matrix is symmetric, only the lower- or uppertriangle is available. After

specifying INCOMPLETE followed by LOWERTRIANGLE or UPPERTRIANGLE the user enters the elements of the supermatrix (20) accordingly, simultaneously taking into account the ROWS/COLUMNS terminal and the DIAGONAL/MISSING terminal. The number of matrix elements which are to be entered depends on which terminal symbols are specified and may be read from Table 1.

Table 1.

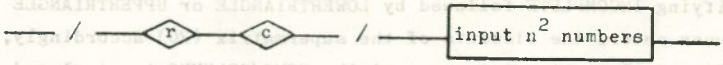
Number of matrix elements

as a function of terminal symbols

Terminal symbols		Number of matrix elements	Explanation
DIAGONAL	COMPLETE	$(kn)^2$	all elements of the matrix (20).
DIAGONAL	INCOMPLETE	$\frac{1}{2}$ kn (kn + 1)	elements in lower- or upper triangle of (20) including kn main diagonal elements.
MISSING	COMPLETE	-	error: no matrix is complete if diagonal elements are missing.
MISSING	INCOMPLETE	$\frac{1}{2}$ kn (kn - 1)	elements in lower- or upper triangle of (20) without kn main diagonal elements.

subarrays

The syntax diagram of the nonterminal subarrays is given in diagram (22). Either $\frac{1}{2}$ k (k + 1) (DIAGONAL given) or $\frac{1}{2}$ k (k - 1) (MISSING given) matrices having n^2 elements are expected. The particular inner product matrix which is entered is specified by the two integers r (for row) and c (for column). Following r and c the elements of the inner product matrix r'c of the super matrix (20) are entered. It is the users responsibility to ensure that all relevant inner product matrices are entered, while the order of entering the inner product matrices is immaterial.



(22)

If DIAGONAL given then $\frac{1}{2}$ k (k + 1) times
else $\frac{1}{2}$ k (k - 1) times.

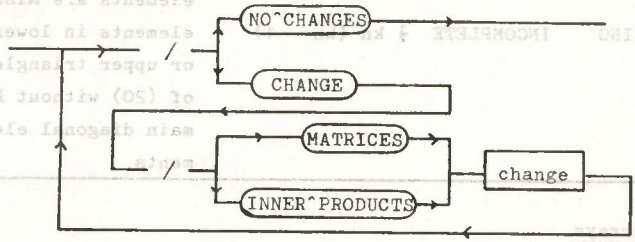
However, it is important to note that the program checks the relation between r and c. When DIAGONAL was specified r must be at least c, while r must be greater than c when MISSING was specified.

transform

After the input of the data matrices or their inner product matrices, the matrices may be transformed in several ways. For example, the user has raw-data matrices but matrices with standard scores are wanted or, in order to solve problem 5, the matrices must be orthonormalized. These and other transformations of the matrices are requested within the transform section.

From the transform syntax diagram (23) it can be seen that the terminal symbol NO CHANGES always appears as the final transform terminal symbol. On the other hand, if the terminal symbol

(23)



CHANGE is given, then the program expects further transform commands. The terminal symbol CHANGE must be followed by the terminal MATRICES (i.e., the data matrices will be transformed) or by the terminal symbol INNER PRODUCTS (i.e., the inner product matrices will be transformed).

Transform commands on MATRICES produce different results than transform commands on INNER PRODUCTS. Table 2 shows the effect of the different transform command combinations.

Table 2

transform command

combinations

Available matrices[a]	terminal symbol in transform	effect
data	MATRICES	Data matrices are transformed before they are printed (cf. the effect of PRE VARIMAX).
inprods	MATRICES	error: the data matrices are not available, so they can not be transformed.
data	INNER PRODUCTS	Data matrices are not transformed before they are printed, but the transformation matrices T_i (i = 1 .. k) are prepared in such a way that $T_i'A_i'A_jT_j$ represents the inner product matrix of matrices i and j after the required transformation.
inprods	INNER PRODUCTS	Inner product matrices are transformed before they are printed.

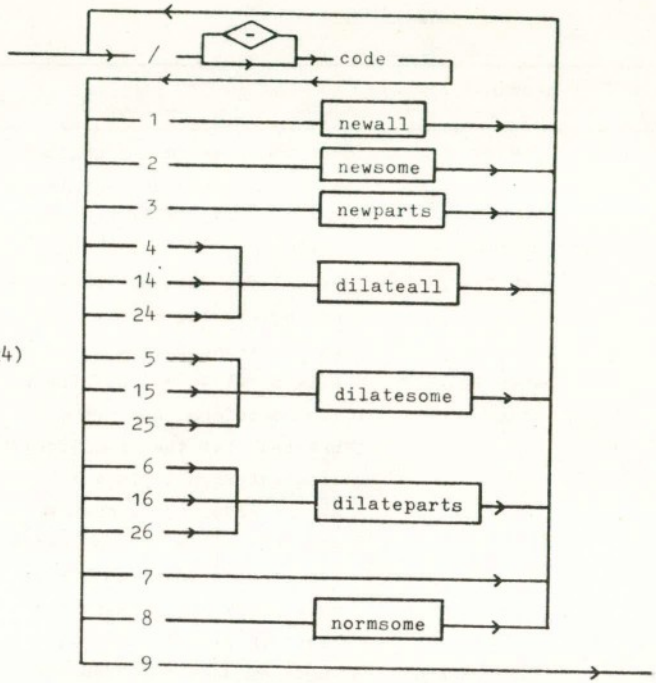
[a]data: data matrices available (i.e., terminal symbol following STARTRUN is MATRICES)

inprods: inner product matrices available (i.e., terminal symbol following STARTRUN is INNER PRODUCTS).

change

The specific change commands are entered in the change section. Depending on the terminal symbol MATRICES or INNER PRODUCTS of the transform section either data matrices or their inner product matrices will be changed (as defined in Table 2), but not both.

The user requests a particular change operation by presenting a change-code, generally followed by additional information (cf. sections newall until normsome). From the change syntax diagram (24) it can be seen that each change command starts on a new



(24)

card or line of text, while the last change code must be code 9, which produces an exit from change. It is important to note that code 9 produces not only an exit from change, but also a return to transform. According to the transform syntax diagram (23) a CHANGE or NO CHANGES terminal symbol follows the change non-terminal symbol. As the last symbol of change is the terminal symbol 9, this 9 is always followed either by CHANGE or by NO CHANGES.

The meaning of all change codes is explained in Table 3.

Table 3

meaning of change codes

APPLICABLE ONLY ON MATRICES	
CHANGING MEANS[a]	
code	effect
1	all columns of all matrices will get one new mean (cf. section newall).
2	all columns of specified matrices will get a new mean (possibly different per matrix, cf. section newsome).
3	some columns of specified matrices will get new means (cf. section newparts).
CHANGING STANDARD DEVIATIONS[a]	
code	effect
4, -4	all columns of all matrices will get one new standard deviation (cf. section dilateall).
5, -5	all columns of specified matrices will get a new standard deviation (possibly different per matrix, cf. section dilatesome).
6, -6	some columns of specified matrices will get new standard deviations (cf. section dilateparts).
APPLICABLE ON MATRICES AND INNER PRODUCTS	
CHANGING SUMS OF SQUARES[b]	
code	effect
14,-14	all columns of all matrices will get one new sum of squares (cf. section dilateall).
24	the total sum of squares of all elements of all matrices will be changed (cf. section dilateall).
15,-15	all columns of specified matrices will get a new sum of squares (possibly different per matrix, cf. section dilatesome).
25	the total sum of squares of all elements of specified matrices will be changed (possibly different per matrix, cf. section dilatesome).

(Continued)

Table 3 - Continued

code	effect
16,-16,26	the sum of squares of all elements of some columns of specified matrices will be changed (cf. section dilateparts).
DIVISION OF MATRIX ELEMENTS[b]	
code	effect
-24	all elements of all matrices will be divided by one number (cf. section dilateall).
-25	all elements of specified matrices will be divided by a number (possibly different per matrix, cf. section dilatesome).
-26	all elements of some columns of some matrices will be divided by a number (cf. section dilateparts).
ORTHONORMALIZATION OF MATRICES[c,b,a]	
code	effect
7	all (inner product) matrices are orthonormalized.
8	specified (inner product) matrices are orthonormalized (cf. section normsome).

[a] As a side-effect sums of squares may be changed

[b] As a side-effect means and standard deviations may be changed

[c] Whenever matrices have linearly dependent columns, the program will stop its run. Whenever inner product matrices have linearly dependent columns, the computer system will abort the COMMA-run and produce a numerical Algol dump.

newall

All columns of all matrices will get the new mean 'const' of the newall syntax diagram (25).

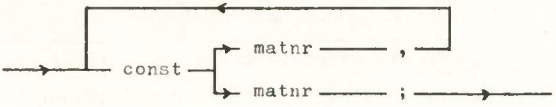
(25)

newsome

From the newsome syntax diagram (26) it can be seen that the

user must provide the program with pairs of numbers. The first of these two numbers ('const') represents the new mean of all columns of the matrix 'matnr' which is the second number of each pair.

(26)

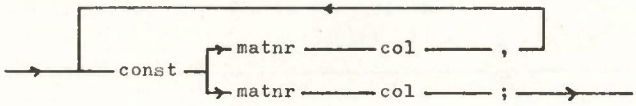


If 'matnr' is immediately followed by a comma, then the program expects another pair of numbers. The last 'matnr' should be followed by a semicolon.

newparts

In newparts ordered triplets are expected as can be seen from the newparts syntax diagram (27). Column 'col' of matrix 'matnr' will get the new mean 'const'. If 'col' is immediately

(27)



followed by a comma, then the program expects a new triplet of numbers. The last 'col' should be followed by a semicolon.

dilateall

The syntax diagram of dilateall (25) shows only one value: 'const'. Depending on the change code 'const' will be interpreted differently. Table 4 shows the different interpretations of 'const'.

dilatesome

The value 'const' in the syntax diagram of dilatesome (26) will also be interpreted differently, depending on the change code. However, instead of changing all columns of all matrices, dilatesome will only change all columns of specified matrices. The six dilatesome change codes 5, 15, 25, -5, -15, -25 correspond to respectively the six dilateall change codes 4, 14, 24, -4, -14 and -25. Consequently, the interpretation column of Table 4 may easily be adapted to dilatesome by changing 'all matrices' into 'the specified matrix'.

Table 4

different interpretations of 'const'

change code	interpretation
4	the standard deviations of all columns of all matrices will be made equal to 'const'.
14	the sums of squares of all columns of all matrices will be made equal to 'const'.
24	the total sum of squares of all elements of all matrices will be made equal to 'const'.
-4	the standard deviations of all columns of all matrices will be made equal to 1/const.
-14	the sums of squares of all columns of all matrices will be made equal to 1/const.
-24	all elements of all matrices will be divided by 'const'.

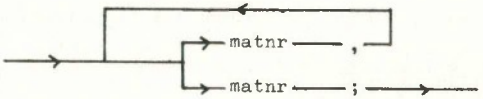
dilateparts

The interpretation of the value 'const' in the syntax diagram of dilateparts (27) depends also on the change code. Dilateparts will only change specified columns of specified matrices. The six dilateparts change codes 6, 16, 26, -6, -16, -26 correspond to respectively the six dilateall change codes 4, 14, 24, -4, -14 and -24. Consequently, the interpretation column of Table 4 may be adapted to dilateparts by changing 'all columns' into 'the specified columns', 'all matrices' into 'the specified matrices' and 'all elements' into 'all elements of the specified columns'.

normsome

As can be seen from the normsome syntax diagram (28), the user provides the program with numbers ('matnr') separated by

(28)



commas except for the last number, which should be followed by a semicolon. The matrices indicated by the numbers will be orthonormalized.

Error Handling

In its current version, COMMA performs several checks on the input data. The program counts the number of terminal symbols which appear in the syntax diagrams in rectangles having rounded angles and calls them "keywords". After an error, COMMA issues a message in which the number of the last keyword is given and in which an errornumber, indicating the kind of error is given. E.g.,

AFTER KEYWORD 4 ERROR 18 HAS BEEN ENCOUNTERED.

The definition of the errornumbers is given in Table 5.

Table 5
errornumbers and
errorcodes

errornumber	errorcode
1	keyword PARAMETERS not found
2	changing standard deviations is impossible if there are only inner product matrices
3	no orthonormalization possible: data columns are linearly dependent
4	no orthonormalization possible: diagonal inner product matrices are missing (normall)
5	equal to error 4, but encountered in the normsome section
6	datamatrices cannot be changed if there are only inner product matrices
7	first matrixnumber may not be less than the second matrixnumber
8	after COMPLETE a complete matrix is expected, the previously given keyword MISSING, however, suggests an incomplete matrix

(Continued)

Table 5 - Continued

errornumber	errorcode
9	in section newsome or newparts: matrix number out of range
10	in section newparts: column number out of range
11	in section dilatesome or dilateparts: matrix number out of range
12	in section dilateparts: column number out of range
13	in section normsome: matrix number out of range
14	in section change: change-code out of range
15	on channel number 6: erroneous data format found
16	on channel number 66: erroneous data format found
17	on channel number 6: end of information prematurely reached
18	on channel number 66: end of information prematurely reached
19	division by zero in section dilateall, dilatesome or dilateparts

Error processing

Currently, COMMA serves two functions. It both checks the input for syntactical errors and, if possible, it processes the input. These two functions will be separated in the future when a separate syntax analyzer becomes available which will perform a complete syntax-check on all input data. However, the user will notice the existence of the syntax-analyzer only from his dayfile or (if COMMA was given input containing errors) from the printed information about the nature and location of the error(s). In other words: the way to start COMMA will not be affected by the presence of the syntax analyzer.

References

ten Berge, J.M.F., Optimizing factorial invariance. Unpublished doctoral dissertation, University of Groningen, 1977a.

ten Berge, J.M.F., Orthogonal procrustes rotation for two and more matrices. Psychometrika, 1977b, 42, 267 - 276.

Cliff, N. & Crus, D.J., Interpretation of canonical analysis: rotated vs. unrotated solutions. Psychometrika, 1976, 41, 35 - 42.

Tucker, L.R., A method for synthesis of factor analytical studies. Personnel Research Section Report No. 984. Washington D.C., Dept. of the army, 1951.