# Ven der Giessen's reordering algorithm in the program for stochastic simulation of econometric models

Bianchi, Carlo and Calzolari, Giorgio and Doret, Remi

IBM Scientific Center, Pisa, Italy

December 1978

IBM Italy
PISA SCIENTIFIC CENTER
Z513-5101
December 1978

# VAN DER GIESSEN'S REORDERING ALGORITHM IN THE PROGRAM FOR STOCHASTIC SIMULATION OF ECONOMETRIC MODELS

Carlo Bianchi — IBM Pisa Scientific Center
Giorgio Calzolari — IBM Pisa Scientific Center
Rémi Doret — Student at the «Institut National Polytechnique», Grenoble

IBM Scientific Center
Via S. Maria 67
56100 Pisa, Italy

This paper describes the application of a reordering algorithm to the equations of econometric models. The algorithm was proposed in 1970 by Van der Giessen [5] and is here applied to the equation format required by the program for stochastic simulation developed at the IBM Scientific Center in Pisa [1], [2].

CONTENTS

1. INTRODUCTION.....p. 7

2. INPUT REQUIREMENTS AND EXECUTION PROCEDURE.....p. 8

2.1. Input data set.....p. 8

2.2. Execution procedure and example of input data set:
the Klein-Goldberger model.....p. 9

2.3. The output data set.....p.11

2.4. Example of output data set:
the reordered Klein-Goldberger model.....p.12

3. PERFORMANCES.....p.14

4. THE PL/1 PROGRAM.....p.15

4.1. A simplified flow-chart.....p.16

4.2. Program listing.....p.18

REFERENCES.....p.26

## 1. INTRODUCTION

The program for stochastic simulation of econometric models proposed in [1] uses the Gauss-Seidel solution algorithm, but does not perform any reordering of the model's equations [2,p.8].

In this paper a program to reorder the equations is proposed. It must be used as a pre-processor; its input data set is the card deck (or disk file) containing the equations of the model, as described in [2,pp.8-9]; the output is a new card deck (or disk file) containing the same equations in a new order. The output data set must then be compiled and successively used as in [2].

The input requirements and the execution procedure are described in some details in section (2.), together with an example on the Klein-Goldberger model. A short comment on the performances of the algorithm is given in section (3.).

Details on the algorithm can be found in the paper by Van der Giessen [5]. In any case, several comment lines are inserted in the PL/1 program, whose listing is printed in section (4.2.), after a simplified flow-chart.

## 2. INPUT REQUIREMENTS AND EXECUTION PROCEDURE

### 2.1. Input data set

It is a card image file.

Cards must be written in a standard FORTRAN code; in particular:

- FORTRAN statements must be written one per card within columns 7 through 72.
- A statement may be continued on successive cards (up to 19) by placing any character other than blank in column 6 of each continuation card.
- On the first card of a statement column 6 must be blank.
- For comments, the character C is placed in column 1.

** Statements allowed

- Statements of the form:

  Y(p)=F(Y(1),....,Y(n),....)

  From now on these statements will be called equations. The subscript of an equation is the first number in parentheses (here p).

  The subscript of the variable Y(n) is n.

- IF statements

  IF (test) followed by an equation

  For example:

  IF(Y(7).GT.Y(10))Y(7)=Y(10)

  The subscript of the IF statement is the subscript of the equation.

** Other requirements

- Variables to be ordered are always of the form Y(...).

  Their subscripts form a sequence 1,2,3,...,n with n up to 500.

- Other character strings including 'Y' in their expressions may be used in the

data.

- Blanks may be present everywhere in a card (except in column 7 for statements).
- There are no restrictions about the number of comment cards.
- Several equations and IF statements may have the same subscript.

## 2.2. Execution procedure and example of input data set: the Klein-Goldberger model

In this section the FORTRAN code for the Klein-Goldberger revised model [4] is displayed according to the format in [2,pp.8-9]. This card-image data set is the 12-th file on the tape whose contents are described in [2].

```
      SUBROUTINE MODEL (Y,X,NEXO,IREAD,IC,YL,NEND,UMC,A)                KLE00010
      IMPLICIT REAL*8 (A-H,O-Z)                                         KLE00020
      DIMENSION Y(1),X(NEXO,IREAD),YL(NEND,IREAD),UMC(1),A(1)           KLE00030
C                                                                       KLE00040
C  MODEL KLEIN-GOLDBERGER ECONOMETRICA , 1969, APRIL                    KLE00050
C                                                                       KLE00060
C  MEANING OF ENDOGENOUS VARIABLES Y(I)                                 KLE00070
C                                                                       KLE00080
C  Y( 1) = CD                                                           KLE00090
C  Y( 2) = CN                                                           KLE00100
C  Y( 3) = RR                                                           KLE00110
C  Y( 4) = HH                                                           KLE00120
C  Y( 5) = IM                                                           KLE00130
C  Y( 6) = X                                                            KLE00140
C  Y( 7) = H                                                            KLE00150
C  Y( 8) = WW                                                           KLE00160
C  Y( 9) = W                                                            KLE00170
C  Y(10) = R                                                            KLE00180
C  Y(11) = I                                                            KLE00190
C  Y(12) = D                                                            KLE00200
C  Y(13) = RS                                                           KLE00210
C  Y(14) = PC                                                           KLE00220
C  Y(15) = NW                                                           KLE00230
C  Y(16) = Y                                                            KLE00240
C  Y(17) = P                                                            KLE00250
C  Y(18) = SC                                                           KLE00260
C  Y(19) = PGR                                                          KLE00270
C  Y(20) = PGRR                                                         KLE00280
C                                                                       KLE00290
C  MEANING OF EXOGENOUS VARIABLES X(I,IC)                               KLE00300
C                                                                       KLE00310
C  X(1,IC) = WG                                                         KLE00320
C  X(2,IC) = PM                                                         KLE00330
C  X(3,IC) = NG                                                         KLE00340
C  X(4,IC) = NS                                                         KLE00350
C  X(5,IC) = NL                                                         KLE00360
C  X(6,IC) = TC                                                         KLE00370
C  X(7,IC) = DU                                                         KLE00380
C  X(8,IC) = RD                                                         KLE00390
C  X(9,IC) = RE-1 (ONLY LAGGED IN THE MODEL)                            KLE00400
C  X(10,IC) = G+E                                                       KLE00410
C  X(11,IC) = TI                                                        KLE00420
```

```
C  X(12,IC) = T                                                          KLE00430
C  X(13,IC) = SUM                                                        KLE00440
C  X(14,IC) = CONSTANT                                                   KLE00450
C                                                                        KLE00460
C  INVESTMENT FUNCTION (NONRESIDENTIAL)  3.14                            KLE00470
      Y(11)=.95*YL(11,IC-1)+A(44)*(YL(6,IC-1)-X(1,IC-1))+               KLE00480
     -A(45)*YL(10,IC-1)+A(46)*YL(11,IC-1)+A(47)*X(14,IC)+UMC(14)        KLE00490
C  DEPRECIATION EQUATION  3.15                                           KLE00500
      Y(12)=A(48)*X(13,IC)+A(49)*X(7,IC)+A(50)*X(14,IC)+UMC(15)         KLE00510
C  INTEREST RATE DETERMINATION EQUATION  3.16                            KLE00520
      Y(13)=A(51)*X(8,IC)+A(52)*X(9,IC)+A(53)*X(7,IC)+                  KLE00530
     -A(54)*X(14,IC)+UMC(16)                                            KLE00540
C  INTEREST RATE STRUCTURE EQUATION  3.10                                KLE00550
      Y(10)=A(31)*Y(13)+A(32)*YL(10,IC-1)+A(33)*X(14,IC)+UMC(10)        KLE00560
C  NATIONAL INCOME NATIONAL PRODUCT IDENTITY  3.18. EXPLIC IN  Y         KLE00570
      Y(16)=(Y(17)*Y(6)-Y(12)-X(11,IC)-Y(17)*Y(18)-                     KLE00580
     -X(6,IC)-X(12,IC))/Y(17)                                           KLE00590
C  CONSUMPTION FUNCTION (DURABLES)  3.1                                  KLE00600
      Y(1)=A(1)*(Y(16)-.7*YL(16,IC-1))+(.7+A(2))*YL(1,IC-1)+A(3)        KLE00610
     -*X(14,IC)+UMC(1)                                                  KLE00620
C  CONSUMPTION FUNCTION (NONDURABLES)  3.2                               KLE00630
      Y(2)=A(4)*Y(16)+A(5)*YL(2,IC-1)+A(6)*X(14,IC)+UMC(2)              KLE00640
C  INVESTMENT FUNCTION (RESIDENTIAL)  3.3                                KLE00650
      Y(3)=A(7)*Y(16)+A(8)*YL(10,IC-1)+A(9)*YL(3,IC-1)+A(10)*X(14,IC)   KLE00660
     -+UMC(3)                                                           KLE00670
C  INVESTMENT FUNCTION (INVENTORIES) SCALED BY 10. NORMALIZED FOR H  3.4KLE00680
      Y(4)=(A(11)/10.*(Y(6)-10.*(-YL(4,IC-1)))+A(12)*YL(4,IC-1)         KLE00690
     -+A(13)/10*X(14,IC)+UMC(4)/10.)/(1.+A(11))                         KLE00700
C  IMPORT DEMAND FUNCTION  3.5                                           KLE00710
      Y(5)=A(14)*Y(6)+A(15)*(X(2,IC)-Y(17))+A(16)*YL(5,IC-1)+           KLE00720
     -A(17)*X(14,IC)+UMC(5)                                             KLE00730
C  DEFINITION OF REAL GNP  3.17                                          KLE00740
      Y(6)=(Y(1)+Y(2)+Y(11)+Y(3)                                        KLE00750
     -+10.*(Y(4)-YL(4,IC-1))                                            KLE00760
     -+X(10,IC)-Y(5))                                                   KLE00770
C  PRODUCTION FUNCTION WITH SCALING BY 100 IN H. EXPLIC. FOR NW  3.6     KLE00780
C  (THE FACTOR 0.230 HAS BEEN EMPIRICALLY CHOSEN FOR CONVERGENCE).       KLE00790
      Y(15)=Y(15)+0.230*                                                KLE00800
     -(Y(6)-(X(1,IC)+.95*(YL(6,IC-1)-X(1,IC-1))+A(18)*(Y(11)+Y(3))      KLE00810
     -+A(19)*((Y(15)-X(3,IC)+X(4,IC))-.95*(YL(15,IC-1)-X(3,IC-1)        KLE00820
     -+X(4,IC-1)))+A(20)/100.*(Y(7)-.95*YL(7,IC-1))+A(21)*X(14,IC)+     KLE00830
     -UMC(6)))                                                          KLE00840
C  HOURS WORKED FUNCTION SCALED BY 100.  3.7                             KLE00850
      Y(7)=A(22)*100.*(Y(9)-YL(9,IC-1))+A(23)*100.*(X(5,IC)-Y(15)-X(4,ICKLE00860
     -))+A(24)*100.*X(14,IC)+UMC(7)*100.                                KLE00870
C  LABOR DEMAND FUNCTION (WAGE SHARE)  3.8                               KLE00880
      Y(8)=X(1,IC)+A(25)*(Y(6)-X(1,IC))+A(26)*(YL(8,IC-1)-X(1,IC-1))    KLE00890
     -+A(27)*X(14,IC)+UMC(8)                                            KLE00900
C  WAGE RATE DETERMINATION EQUATION  3.9                                 KLE00910
      Y(9)=YL(9,IC-1)+A(28)*(X(5,IC)-Y(15)-X(4,IC))+                    KLE00920
     -A(29)*(YL(17,IC-1)-YL(17,IC-2))+A(30)*X(14,IC)+UMC(9)             KLE00930
C  WAGE IDENTITY NORMALIZED FOR P 3.20                                   KLE00940
      Y(17)=Y(7)*Y(9)*Y(15)/Y(8)/100.                                   KLE00950
C  CORPORATE SAVINGS FUNCTION SCALED BY 10.   3.11. EXPLIC. IN  SC       KLE00960
      Y(18)=(A(34)*(Y(17)*Y(14)-X(6,IC))+A(35)*(YL(17,IC-1)*YL(14,IC-1) KLE00970
     --X(6,IC-1)-YL(17,IC-1)*YL(18,IC-1))+A(36)*X(14,IC)+UMC(11))/      KLE00980
     -Y(17)                                                             KLE00990
C  RENTIER INCOME EQUATION  3.13. EXPLIC IN  PGRR                        KLE01000
      Y(20)=(A(40)*Y(17)*(Y(11)+Y(3))+A(41)*(Y(10)-YL(10,IC-1))         KLE01010
     -+A(42)*YL(20,IC-1)*YL(17,IC-1)+A(43)*X(14,IC)+UMC(13))/Y(17)      KLE01020
C  DEFINITION OF PROFIT  3.19. EXPLIC. IN  PGR                           KLE01030
      Y(19)=(Y(17)*Y(6)-Y(12)-X(11,IC)-Y(17)*Y(8)-Y(20)*Y(17))/Y(17)    KLE01040
C  NONCORPORATE INCOME EQUATION  3.12. EXPLIC. IN  PC                    KLE01050
      Y(14)=Y(19)-10./Y(17)*                                            KLE01060
     -(A(37)/10.*Y(17)*Y(6)+A(38)*(YL(19,IC-1)-YL(14,IC-1))*YL(17,IC-1)/KLE01070
     -10.                                                               KLE01080
     -+A(39)/10.*X(14,IC)+UMC(12)/10.)                                  KLE01090
C*********************************************************************KLE01100
      RETURN                                                            KLE01110
      END                                                               KLE01120
```

To process the reordering algorithm, first of all the first 46 cards of the data set must be dropped, stored into a separate file, as well as the RETURN and END statements. The comment card

C**********

is mandatory at the end of the data set and must appear nowhere else.

After these preliminary operations, INPUT FORTRAN A1 must be the CMS identifier of the data set [3]; the following VANDERG EXEC procedure

```
FILEDEF ONE DISK INPUT FORTRAN A1 (RECFM F LRECL 80 BLOCK 80)
FILEDEF TWO DISK OUTPUT FORTRAN A1 (RECFM F LRECL 80 BLOCK 80)
GLOBAL TXTLIB PLILIB
LOAD VANDERG
START
```

can now be used issuing the CMS command:

```
vanderg
```

## 2.3. The output data set

The output processing is performed by the subroutines ECRIT and IMPRIM (see section 4.). The equations are written following the sequence (put in SEQ) found by Van der Giessen's method.

All the comments preceding a statement in the input file are present in the output file immediately before the statement itself.

Several statements may have the same subscript. In the output file they are written in a block (inserted at the place indicated by SEQ). Inside the block they have the same sequence as their relative order in the input file.

The comment 'END OF RECURSIVE PART' is written after the end of the recursive equations (determined in subroutine RNG), if any.

The comment 'END OF SIMULTANEOUS PART' is put just before the beginning of the post-recursive part (if any).

For further details see the program listing (MAIN and subroutine RNG, section 4.2.).

2.4. Example of output data set:

the reordered Klein-Goldberger model

The result of the execution of the VANDERG EXEC procedure is the creation of the file whose CMS identifier is OUTPUT FORTRAN, containing the reordered equations and the corresponding comment lines as follows:

```
C  INVESTMENT FUNCTION (NONRESIDENTIAL)  3.14                              OUT00010
      Y(11)=.95*YL(11,IC-1)+A(44)*(YL(6,IC-1)-X(1,IC-1))+                  OUT00020
     +A(45)*YL(10,IC-1)+A(46)*YL(11,IC-1)+A(47)*X(14,IC)+UMC(14)           OUT00030
C  DEPRECIATION EQUATION  3.15                                             OUT00040
      Y(12)=A(48)*X(13,IC)+A(49)*X(7,IC)+A(50)*X(14,IC)+UMC(15)            OUT00050
C  INTEREST RATE DETERMINATION EQUATION  3.16                              OUT00060
      Y(13)=A(51)*X(8,IC)+A(52)*X(9,IC)+A(53)*X(7,IC)+                     OUT00070
     +A(54)*X(14,IC)+UMC(16)                                               OUT00080
C  INTEREST RATE STRUCTURE EQUATION  3.10                                  OUT00090
      Y(10)=A(31)*Y(13)+A(32)*YL(10,IC-1)+A(33)*X(14,IC)+UMC(10)           OUT00100
C****     END OF RECURSIVE PART     *****                                 OUT00110
C  WAGE RATE DETERMINATION EQUATION  3.9                                   OUT00120
      Y(9)=YL(9,IC-1)+A(28)*(X(5,IC)-Y(15)-X(4,IC))+                       OUT00130
     +A(29)*(YL(17,IC-1)-YL(17,IC-2))+A(30)*X(14,IC)+UMC(9)                OUT00140
C  HOURS WORKED FUNCTION SCALED BY 100.  3.7                               OUT00150
      Y(7)=A(22)*100.*(Y(9)-YL(9,IC-1))+A(23)*100.*(X(5,IC)-Y(15)-X(4,ICOUT00160
     +))+A(24)*100.*X(14,IC)+UMC(7)*100.                                   OUT00170
C  INVESTMENT FUNCTION (INVENTORIES) SCALED BY 10. NORMALIZED FOR H  3.4OUT00180
      Y(4)=(A(11)/10.*(Y(6)-10.*(-YL(4,IC-1)))+A(12)*YL(4,IC-1)           OUT00190
     ++A(13)/10*X(14,IC)+UMC(4)/10.)/(1.+A(11))                            OUT00200
C  LABOR DEMAND FUNCTION (WAGE SHARE)  3.8                                 OUT00210
      Y(8)=X(1,IC)+A(25)*(Y(6)-X(1,IC))+A(26)*(YL(8,IC-1)-X(1,IC-1))       OUT00220
     ++A(27)*X(14,IC)+UMC(8)                                               OUT00230
C  WAGE IDENTITY NORMALIZED FOR P 3.20                                     OUT00240
      Y(17)=Y(7)*Y(9)*Y(15)/Y(8)/100.                                      OUT00250
C  IMPORT DEMAND FUNCTION  3.5                                             OUT00260
      Y(5)=A(14)*Y(6)+A(15)*(X(2,IC)-Y(17))+A(16)*YL(5,IC-1)+              OUT00270
     +A(17)*X(14,IC)+UMC(5)                                                OUT00280
C  DEFINITION OF PROFIT  3.19. EXPLIC. IN  PGR                             OUT00290
      Y(19)=(Y(17)*Y(6)-Y(12)-X(11,IC)-Y(17)*Y(8)-Y(20)*Y(17))/Y(17)       OUT00300
C  NONCORPORATE INCOME EQUATION  3.12. EXPLIC. IN  PC                      OUT00310
      Y(14)=Y(19)-10./Y(17)*                                               OUT00320
     +(A(37)/10.*Y(17)*Y(6)+A(38)*(YL(19,IC-1)-YL(14,IC-1))*YL(17,IC-1)/OUT00330
     +10.                                                                  OUT00340
     ++A(39)/10.*X(14,IC)+UMC(12)/10.)                                     OUT00350
C  CORPORATE SAVINGS FUNCTION SCALED BY 10.    3.11. EXPLIC. IN  SC        OUT00360
      Y(18)=(A(34)*(Y(17)*Y(14)-X(6,IC))+A(35)*(YL(17,IC-1)*YL(14,IC-1) OUT00370
     +-X(6,IC-1)-YL(17,IC-1)*YL(18,IC-1))+A(36)*X(14,IC)+UMC(11))/        OUT00380
     +Y(17)                                                                OUT00390
C  NATIONAL INCOME NATIONAL PRODUCT IDENTITY  3.18. EXPLIC IN  Y           OUT00400
      Y(16)=(Y(17)*Y(6)-Y(12)-X(11,IC)-Y(17)*Y(18)-                        OUT00410
     +X(6,IC)-X(12,IC))/Y(17)                                              OUT00420
C  CONSUMPTION FUNCTION (DURABLES)  3.1                                    OUT00430
      Y(1)=A(1)*(Y(16)-.7*YL(16,IC-1))+(.7+A(2))*YL(1,IC-1)+A(3)           OUT00440
     +*X(14,IC)+UMC(1)                                                     OUT00450
C  CONSUMPTION FUNCTION (NONDURABLES)  3.2                                 OUT00460
      Y(2)=A(4)*Y(16)+A(5)*YL(2,IC-1)+A(6)*X(14,IC)+UMC(2)                 OUT00470
C  INVESTMENT FUNCTION (RESIDENTIAL)  3.3                                  OUT00480
      Y(3)=A(7)*Y(16)+A(8)*YL(10,IC-1)+A(9)*YL(3,IC-1)+A(10)*X(14,IC)      OUT00490
     ++UMC(3)                                                              OUT00500
C  DEFINITION OF REAL GNP  3.17                                            OUT00510
      Y(6)=(Y(1)+Y(2)+Y(11)+Y(3)                                           OUT00520
     ++10.*(Y(4)-YL(4,IC-1))                                               OUT00530
     ++X(10,IC)-Y(5))                                                      OUT00540
C  PRODUCTION FUNCTION WITH SCALING BY 100 IN H. EXPLIC. FOR NW  3.6       OUT00550
C  (THE FACTOR 0.230 HAS BEEN EMPIRICALLY CHOSEN FOR CONVERGENCE).         OUT00560
```

```
      Y(15)=Y(15)+0.230*                                                OUT00570
     +(Y(6)-(X(1,IC)+.95*(YL(6,IC-1)-X(1,IC-1))+A(18)*(Y(11)+Y(3))      OUT00580
     ++A(19)*((Y(15)-X(3,IC)+X(4,IC))-.95*(YL(15,IC-1)-X(3,IC-1)        OUT00590
     ++X(4,IC-1)))+A(20)/100.*(Y(7)-.95*YL(7,IC-1))+A(21)*X(14,IC)+     OUT00600
     +UMC(6)))                                                         OUT00610
C  RENTIER INCOME EQUATION  3.13. EXPLIC IN  PGRR                       OUT00620
      Y(20)=(A(40)*Y(17)*(Y(11)+Y(3))+A(41)*(Y(10)-YL(10,IC-1))        OUT00630
     ++A(42)*YL(20,IC-1)*YL(17,IC-1)+A(43)*X(14,IC)+UMC(13))/Y(17)     OUT00640
```

The heading statements previously dropped, as well as the RETURN and END statements, must be reinserted.

The data set so obtained (OUTPUT FORTRAN) must be renamed KLEINGOL FORTRAN and compiled; simulation can then be performed as in [2,p.29].

## 3. PERFORMANCES

The solution of the non-reordered Klein-Goldberger model for one year (for example 1965), with a relative tolerance 0.1E-06 and assuming the historical values of the endogenous variables as starting values of the Gauss-Seidel procedure, requires 42 iterations.

The solution of the model, reordered as in section (2.4.), requires 18 iterations.

The solution of the Klein-I model, as in [2,pp.8-9], at 1941 requires 25 iterations. After reordering, the same model requires 18 iterations.

The reordering of the Klein-Goldberger model requires about 8 seconds of CPU time on a computer IBM/370 model 168.

## 4. THE PL/1 PROGRAM

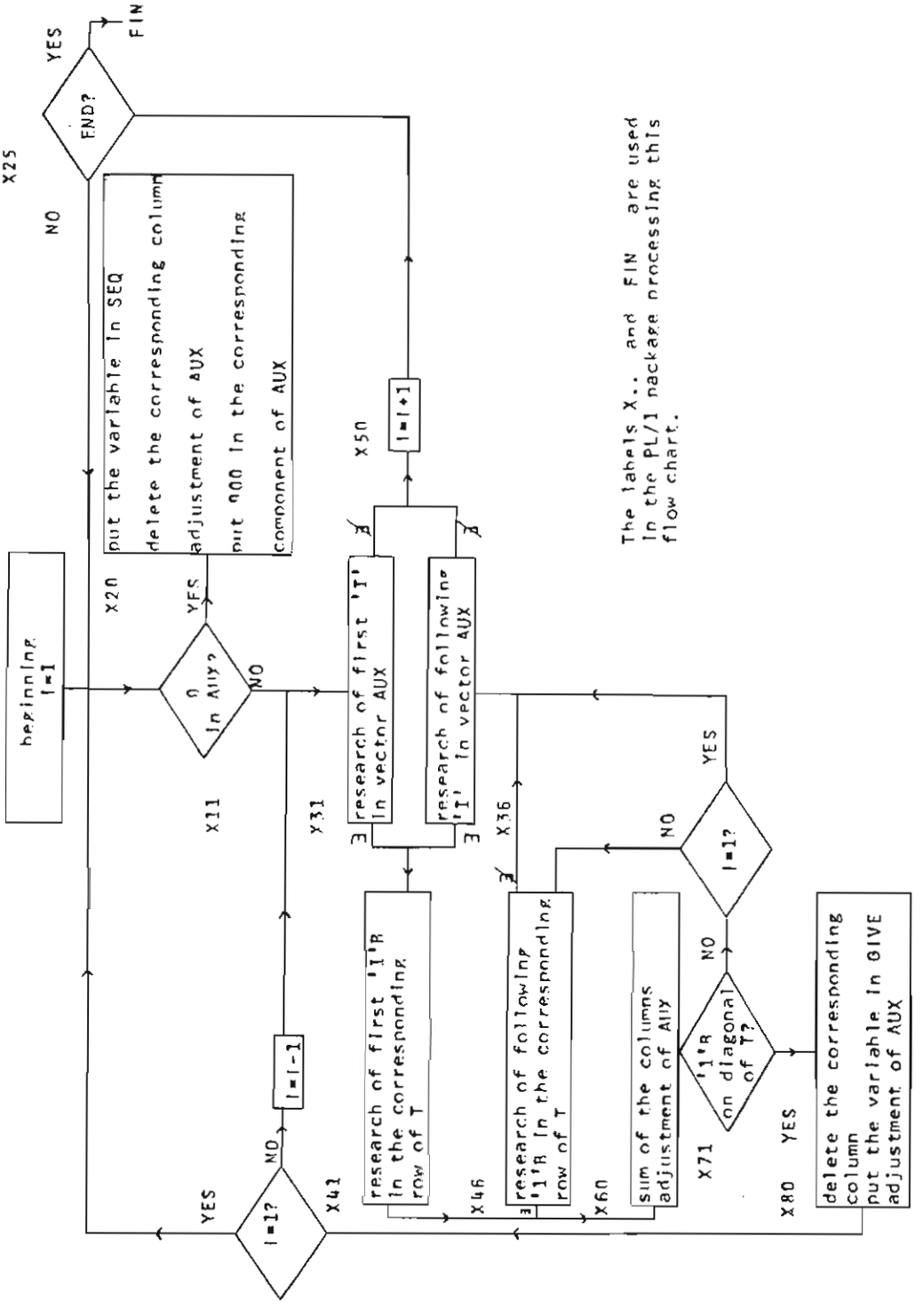
Whenever possible, within the program listing and in the flow-charts the same variables names as in Van der Giessen's paper [5] are maintained.

The program:

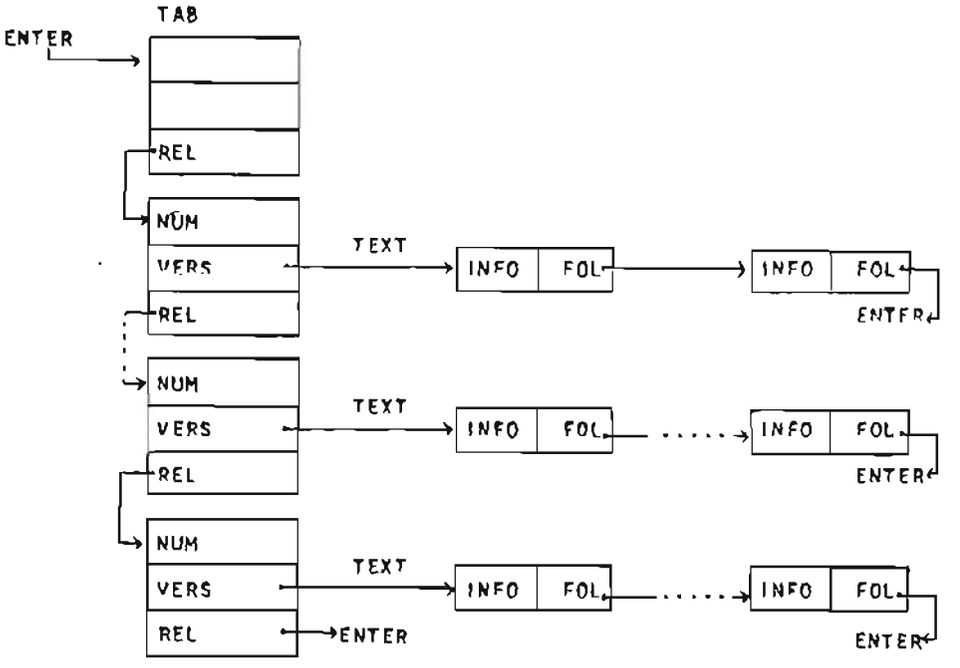
- reads the data set,
- fills the structure matrix T after a syntactic analysis (research of the sequence Y(...)) of each equation,
- stores the cards in a lists structure (see later in this paragraph),
- processes the Van der Giessen's algorithm,
- fills the output file with the new sequence.

Details about each subroutine are given in the listing of the program (see 4.2.).

4.1. A simplified flow-chart



In order to have a simple output processing, the cards are stored in this way (a list of lists is used):




The first element of TAB is just created to begin the main list.

In each list TEXT we have a statement (equation or IF statement) with the comment cards coming just before this statement. Each INFO contains one card of the statement. NUM is the subscript of the equation or IF statement.

If several statements have the same subscript, the corresponding row in T depends on all the variables found in all the statements.

## 4.2. Program listing

```
/* PROPRIETA' DELLA IBM ITALIA  */                                    VAN00010
/* CENTRO SCIENTIFICO DI PISA   */                                    VAN00020
                                                                      VAN00030
        VANDERG: PROC OPTIONS(MAIN);                                  VAN00040
                 /*****************************/                      VAN00050
                 /****                     ****/                      VAN00060
                 /*       DECLARATIONS        */                      VAN00070
                 /****                     ****/                      VAN00080
                 /*****************************/                      VAN00090
                                                                      VAN00100
DCL RANG1 POINTER;                                                    VAN00110
DCL (AUX,GIVE,SEQ)(500)PIC'999';                                      VAN00120
DCL REC (500) PIC'999'; /* USED TO WRITE 'RECURSIVE PART' */          VAN00130
DCL R1 PIC'999' ;       /* POINTER IN VECTOR 'REC'        */          VAN00140
    /* THESE ARRAYS ARE USED IN SUBROUTINE RNG                    */ VAN00150
    /* AUX(I) IS FILLED WHITH THE NUMBER OF '1'S IN ROW I OF MATRIX T*/VAN00160
    /* AT THE END OF VOG ALGORITHM, 'GIVE' IS FILLED WITH THE     */ VAN00170
    /* VARIABLES TO BE GIVEN                                      */ VAN00180
    /* 'SEQ' IS FILLED WITH THE NEW ORDER OF THE EQUATIONS        */ VAN00190
DCL ONE FILE RECORD INPUT;                                            VAN00200
    /* FILE ONE COUNTAINS DATA I.E. THE SET OF EQUATIONS AND COMMENTS*/VAN00210
DCL TWO FILE RECORD OUTPUT;                                           VAN00220
    /*AFTER PROCESSING FILE TWO CONTAINS THE RESULTS OF REORDERING */ VAN00230
DCL COM BIT(1); /* LOGICAL USED FOR THE PRINTING OF COMMENTS */       VAN00240
DCL N PIC'999', /* NUMBER OF EQUATIONS */                             VAN00250
    I PIC'99',  /* POINTER IN LENG     */                            VAN00260
    CARD CHAR(80);   /* CARD USED TO READ DATA */                     VAN00270
DCL Z1 PIC'999' , EQU CHAR(7000) VARYING ;                            VAN00280
    /* BEFORE EACH 'CALL STUD' EQU CONTAINS ONE EQUATION */           VAN00290
DCL  ROW PIC'999',LENG(99) PIC'999',J PIC'9999';                      VAN00300
    /* ROW IS THE INDEX OF ROW IN MATRIX T          */                VAN00310
    /* LENG IS AN ARRAY USED IN SUBROUTINE CREATE  */                 VAN00320
DCL T (500,500 ) BIT;     /* STRUCTURE ARRAY  */                      VAN00330
    /* THE TWO FOLLOWING STRUCTURES FORM LISTS TO STORE EQUATIONS */  VAN00340
DCL 1 TAB BASED (Q),                                                  VAN00350
     2 NUM PIC'999', /* NUM ROW OF AN EQUATION Y(2)=...->NUM=2    */  VAN00360
     2 VERS POINTER, /* USED TO RECOVER THE FIRST ELEMENT OF TEXT */  VAN00370
     2 REL POINTER;  /*  POINT  ON THE FOLLOWING ELEMENT OF TAB   */  VAN00380
DCL 1 TEXT BASED(P),                                                  VAN00390
     2 INFO CHAR(80), /* INFO IS FILLED WITH 'A LINE OF AN EQUATION' */VAN00400
     2 FOL POINTER;   /* POINT ON THE FOLLOWING ELEMENT OF TEXT      */VAN00410
DCL (Q,P) POINTER ;                                                   VAN00420
DCL COUR1 POINTER ; /*  CURRENT POINTER IN LIST TAB   */              VAN00430
DCL COUR2 POINTER ; /*  CURRENT POINTER IN LIST TEXT  */              VAN00440
DCL ENTER POINTER ; /* BEGINNING OF THE LIST TAB      */              VAN00450
                                                                      VAN00460
                 /*****************************/                      VAN00470
                 /****                     ****/                      VAN00480
                 /*       SUBROUTINES       */                      VAN00490
                 /****                     ****/                      VAN00500
                 /*****************************/                      VAN00510
                                                                      VAN00520
/*********************************************************************/VAN00530
/*  AS IN THE DATA AN EQUATION MAY BE GIVEN ON SEVERAL CARDS      */ VAN00540
/*  THIS PROCEDURE PUTS ALL THE CARDS COMPOSING AN EQUATION OR IF */ VAN00550
/*  STATEMENT IN THE CHARACTER STRING CALLED 'EQU'                */ VAN00560
/*  THE ARRAY 'LENG'                                              */ VAN00570
/*  LENG(1) CONTAINS THE ROW OF THE EQUATION Y(4)=...=>LENG(1)=4  */ VAN00580
/*  LENG(I) CONTAINS THE LENGTH (AFTER CANCELLING THE BLANKS)     */ VAN00590
/*  OF THE I-1 CARD OF AN EQUATION    OR  OF AN IF STATEMENT      */ VAN00600
/*********************************************************************/VAN00610
/*-----------------*/                                                 VAN00620
    CREATE:PROC;                                                      VAN00630
/*-----------------*/                                                 VAN00640
DCL K PIC'99';                                                        VAN00650
LENG=0;I=2;K=72;                                                      VAN00660
```

```
     /* AFTER CANCELLING THE BLANKS 'K' IS ON THE LAST NONBLANK  */     VAN00670
     /* CHARACTER OF A CARD                                      */     VAN00680
DO WHILE(SUBSTR(CARD,K,1)=' ');K=K-1;END;                             VAN00690
     /* WE PUT THE FIRST CARD OF THE EQUATION IN 'EQU'           */     VAN00700
EQU=SUBSTR(CARD,7,K-6);                                               VAN00710
LENG(2)=K-6;J=K-6;                                                    VAN00720
READ FILE(ONE) INTO (CARD);                                           VAN00730
I=3;                                                                  VAN00740
      /* THE FOLLOWING LOOP FINDS THE REST OF CARDS COMPOSING AN  */     VAN00750
      /* EQUATION AND PUT THEM IN 'EQU'                           */     VAN00760
DO WHILE((SUBSTR(CARD,1,1)¬='C')&(SUBSTR(CARD,6,1)¬=' '));            VAN00770
  K=72;                                                               VAN00780
  DO WHILE(SUBSTR(CARD,K,1)=' ');K=K-1;END;                           VAN00790
  EQU=SUBSTR(EQU,1,J)||SUBSTR(CARD,7,K-6);                            VAN00800
  LENG(I)=K-6;                                                        VAN00810
  I=I+1;J=J+K-6;                                                      VAN00820
  READ  FILE(ONE) INTO (CARD);                                        VAN00830
  END;                                                                VAN00840
END CREATE;                                                           VAN00850
                                                                      VAN00860
/*************************************************************/       VAN00870
/*  WHEN WE HAVE FOUND A STRING 'Y(...)', 'COD'LOOKS FOR     */       VAN00880
/*  THE INDEX BETWEEN PARENTHESES AND PUT IT IN 'Z1'         */       VAN00890
/*************************************************************/       VAN00900
/*------------------*/                                                VAN00910
    COD: PROC(L);                                                     VAN00920
/*------------------*/                                                VAN00930
DCL Z4 CHAR(2);                                                       VAN00940
DCL (L,K) PIC'9999';                                                  VAN00950
K=L;                                                                  VAN00960
       /* WE LOOK FOR RIGHT PARENTHESIS OF Y(    )                */     VAN00970
DO WHILE(SUBSTR(EQU,K,1)¬=')');K=K+1;END;                             VAN00980
K=K-1;                                                                VAN00990
       /* WE ARE BETWEEN THE PARENTHESES OF Y(  )AND WE LOOK FOR */     VAN01000
       /* THE FIRST DIGIT OF THE INDEX STARTING FROM THE RIGHT   */     VAN01010
DO WHILE(SUBSTR(EQU,K,1)=' ');K=K-1;END;                              VAN01020
       /* WE EXAMINE IF THERE ARE ONE OR TWO DIGITS  IN INDEX    */     VAN01030
       /* AND FILL 'Z1' CONSEQUENTLY                             */     VAN01040
IF((SUBSTR(EQU,K-1,1)=' ')|(SUBSTR(EQU,K-1,1)='('))                   VAN01050
THEN Z1='00'||SUBSTR(EQU,K , 1);                                      VAN01060
ELSE IF((SUBSTR(EQU,K-2,1)=' ')|(SUBSTR(EQU,K-2,1)='('))              VAN01070
      THEN Z1='0'||SUBSTR(EQU,K-1,2);                                 VAN01080
     ELSE Z1=SUBSTR(EQU,K-2,3);                                       VAN01090
END COD;                                                              VAN01100
                                                                      VAN01110
/*************************************************************/       VAN01120
/* THIS SUBROUTINE FILLS A ROW OF STRUCTURE ARRAY 'T'        */       VAN01130
/* THE EQUATION TO DEAL WITH IS IN 'EQU'                     */       VAN01140
/*************************************************************/       VAN01150
/*------------------*/                                                VAN01160
    STUD:PROC(M3);                                                    VAN01170
/*------------------*/                                                VAN01180
DCL(M1,M2,M3)PIC'9999',Z2 CHAR(1),                                    VAN01190
    COL PIC'999';                                                     VAN01200
      /* COD(M3 ) GIVES US THE ROW TO FILL IN MATRIX 'T'        */     VAN01210
CALL COD(M3);ROW=Z1;LENG(1)=ROW;M1=M3+3;                              VAN01220
DO WHILE(M1<J); /* J INDICATES THE END OF 'EQU' */                    VAN01230
IF(SUBSTR(EQU,M1,1)='Y') THEN                                         VAN01240
       /* WE HAVE FOUND THE CHARACTER 'Y',  WE LOOK IF THERE IS   */     VAN01250
       /* THE FOLLOWING SEQUENCE : Y(....)                       */     VAN01260
DO;M2=M1+1;                                                           VAN01270
 DO WHILE (SUBSTR(EQU,M2,1)=' ');M2=M2+1;END;                         VAN01280
  IF( SUBSTR(EQU,M2,1) ='(') THEN                                     VAN01290
       DO    ;                                                        VAN01300
    Z2=SUBSTR(EQU,M1-1,1);                                            VAN01310
   IF((Z2 =' ')|(Z2 ='*')|(Z2 ='+')|(Z2 ='-')|(Z2 ='(')               VAN01320
      |(Z2='/')|(Z2='=')) THEN                                        VAN01330
         /* WE HAVE FOUND A STRING 'Y(...)' IN EQU. 'COD' GIVES THE */VAN01340
         /* COLUMN INDEX 'COL' AND WE DO T(ROW,COL)='1'B            */VAN01350
        DO    ;                                                       VAN01360
```

```
            CALL COD(M1);                                                    VAN01370
            COL=Z1;T(ROW,COL)='1'B;                                          VAN01380
            M1=M1+3;                                                         VAN01390
          END;                                                               VAN01400
        END;                                                                 VAN01410
    END;                                                                     VAN01420
M1=M1+1;                                                                     VAN01430
END;                                                                         VAN01440
END STUD ;                                                                   VAN01450
                                                                             VAN01460
/*************************************************************/             VAN01470
/*'ECRIT' PRINTS THE EQUATION WHOSE INDEX IS ORD(Y(ORD)=....) */             VAN01480
/* IN THE SAME WAY AS IT WAS GIVEN IN DATA                    */             VAN01490
/*************************************************************/             VAN01500
/*-------------------------*/                                                VAN01510
    ECRIT:PROC(ORD,RANG);                                                    VAN01520
/*-------------------------*/                                                VAN01530
DCL RANG POINTER ; /* USED TO PRINT SEVERAL LIST HAVING SAME INDEX */        VAN01540
DCL ORD PIC'999';                                                            VAN01550
COUR1=RANG; /* RESEARCH OF INDEX ORD BEGINS AT ADDRESS RANG*/               VAN01560
DO WHILE((COUR1->NUM^=ORD)&(COUR1^=ENTER));                                  VAN01570
COUR1=COUR1->REL;                                                            VAN01580
END;                                                                         VAN01590
IF(COUR1=ENTER) THEN GOTO TERM;                                              VAN01600
COUR2=COUR1->VERS;                                                           VAN01610
      /* WE ARE ON THE RIGHT ADDRESS,WE WRITE THE FIRST CARD */              VAN01620
 WRITE FILE(TWO) FROM(COUR2->INFO);                                          VAN01630
      /*   THE FOLLOWING LOOP WRITES THE REST OF TEXT    */                  VAN01640
DO WHILE(COUR2->FOL^=ENTER);                                                 VAN01650
  COUR2=COUR2->FOL;                                                          VAN01660
  WRITE FILE(TWO) FROM(COUR2->INFO);                                         VAN01670
END;                                                                         VAN01680
TERM:END ECRIT;                                                              VAN01690
                                                                             VAN01700
/*************************************************************/             VAN01710
/* THIS SUBROUTINE STORES AN EQUATION IN THE SAME WAY AS     */             VAN01720
/* IT WAS READ IN ORDER TO HAVE A SIMPLE PRINTING. THIS IS   */             VAN01730
/* DONE WITH A LIST STRUCTURE                                */             VAN01740
/*************************************************************/             VAN01750
/*-------------*/                                                           VAN01760
    ST:PROC;                                                                 VAN01770
/*-------------*/                                                           VAN01780
DCL CARD1 CHAR(80),                                                          VAN01790
    POS PIC'9999', /* INDICATES POSITION IN STRING 'EQU' */                 VAN01800
    R PIC'99';                                                               VAN01810
    /* PROCESSING OF THE FIRST CARD OF AN EQUATION  OR IF STATEMENT. */      VAN01820
    /* FIRSTLY WE EXAMINE IF COMMENTS HAVE BEEN ALREADY PUT          */      VAN01830
    /* IN A LIST TEXT : IF COM='0'B (NO COMMENTS) WE CREATE A NEW    */      VAN01840
    /* LIST TEXT(Y1) OTHERWISE WE CONTINUE THE LIST ALREADY BEGUN(Y2)*/      VAN01850
CARD1=' ' ;                                                                  VAN01860
CARD1='      '||SUBSTR(EQU,1,LENG(2));                                       VAN01870
IF (COM='0'B ) THEN GOTO Y1;                                                 VAN01880
 ELSE GOTO Y2;                                                               VAN01890
Y2: COUR1->NUM=LENG(1);                                                      VAN01900
    ALLOCATE TEXT SET (P);                                                   VAN01910
    COUR2->FOL=P;                                                            VAN01920
    P->INFO=CARD1;                                                           VAN01930
    P->FOL=ENTER;                                                            VAN01940
    COUR2=P;                                                                 VAN01950
    POS=LENG(2)+1;                                                           VAN01960
    GOTO Y3;                                                                 VAN01970
Y1:ALLOCATE TAB SET (Q);                                                     VAN01980
   COUR1->REL=Q;                                                             VAN01990
   Q->REL=ENTER;                                                             VAN02000
   Q->NUM=LENG(1);                                                           VAN02010
   COUR1=Q;                                                                  VAN02020
   ALLOCATE TEXT SET(P);                                                     VAN02030
   COUR2=P;                                                                  VAN02040
   Q->VERS=P;                                                                VAN02050
   P->INFO=CARD1;                                                            VAN02060
```

```
    P->FOL=ENTER;                                                          VAN02070
    POS=LENG(2)+1;                                                         VAN02080
Y3:R=3;                                                                    VAN02090
       /* LOOP TO CONTINUE THE  LIST 'TEXT' UNTIL THE END OF  */           VAN02100
       /* THE EQUATION                                        */           VAN02110
    DO WHILE (POS<J);                                                      VAN02120
     CARD1=' ';                                                            VAN02130
     CARD1='     '||SUBSTR(EQU,POS,LENG(R));                               VAN02140
     ALLOCATE TEXT SET(P);                                                 VAN02150
     COUR2->FOL=P;                                                         VAN02160
     P->INFO=CARD1;                                                        VAN02170
     P->FOL=ENTER;                                                         VAN02180
     COUR2=P;                                                              VAN02190
     POS=POS + LENG(R);                                                    VAN02200
     R=R+1;                                                                VAN02210
    END;                                                                   VAN02220
END ST ;                                                                   VAN02230
                                                                           VAN02240
/*************************************************************/           VAN02250
/* THIS PROCEDURE RECOGNIZES IF WE  ARE ON A STATEMENT  */                 VAN02260
/* Y( )= OR AN'IF STAT'( IF(.....) ).AFTER THERE        */                 VAN02270
/* IS A SPECIAL TREATMENT FOR EACH KIND OF STATEMENT    */                 VAN02280
/*************************************************************/           VAN02290
/*-----------------*/                                                     VAN02300
    TESTY:PROC ;                                                           VAN02310
/*-----------------*/                                                     VAN02320
DCL PEQU PIC '9999',/* POINTER IN STRING EQU                      */      VAN02330
    NPC  PIC '999'; /* USED TO TEST A NUMBER OF PARENTHESIS       */      VAN02340
IF (SUBSTR(EQU,1,1)¬='Y') THEN                                             VAN02350
       /* WE ARE ON AN 'IF STAT'  */                                       VAN02360
   DO;                                                                     VAN02370
   PEQU=1;                                                                 VAN02380
       /* WE LOOK FOR THE LEFT PARENTHESIS OF Y(  )   */                   VAN02390
   DO WHILE(SUBSTR(EQU,PEQU,1)¬='('); PEQU=PEQU+1;  END;                   VAN02400
       /* PEQU IS ON THE FIRST PARENTHESIS  */                             VAN02410
   NPC=1 ; PEQU = PEQU + 1;                                                VAN02420
   DO WHILE(NPC¬=0) ;                                                      VAN02430
      /* WE LOOK FOR THE END OF TEST IF BY COUNTING THE PARENTHESES */VAN02440
      /* WITH NPC                                                   */VAN02450
     IF (SUBSTR(EQU,PEQU,1)='(') THEN     NPC = NPC +1;                  VAN02460
     ELSE IF (SUBSTR(EQU,PEQU,1)=')') THEN      NPC = NPC -1;              VAN02470
   PEQU=PEQU+1;                                                            VAN02480
   END;                                                                    VAN02490
   PEQU = PEQU +1;                                                         VAN02500
       /* WE LOOK FOR THE FIRST CHAR ¬' ' FOLLOWING THE END OF TEST */VAN02510
    DO WHILE (SUBSTR(EQU,PEQU,1)=' ') ; PEQU = PEQU +1; END;               VAN02520
   IF ((SUBSTR(EQU,PEQU,2)='Y(')|(SUBSTR(EQU,PEQU,2)='Y ')) THEN           VAN02530
       /* THE STAT FOLLOWING IF(..) IS Y( )=..  WE HAVE TO TREAT   */VAN02540
       /* EQUATION IN STUD(PEQU) PEQU IS NOW ON  'Y' OF Y(    )    */VAN02550
     CALL STUD(PEQU);                                                      VAN02560
   ELSE                                                                    VAN02570
        /* THERE IS NOT Y(.)= AFTER 'IF (...)' SO WE HAVE JUST TO */VAN02580
        /* STORE THIS STATE  WITH 999 IN 'NUM' OF LIST 'TAB'      */VAN02590
        /* WHICH MEANS THAT THIS STATE WILL BE PRINTED AT THE END */VAN02600
    LENG(1)=999;                                                           VAN02610
   END;                                                                    VAN02620
ELSE DO;                                                                   VAN02630
       /* THE STATEMENT IS 'Y(..)=.... ' WE TREAT IT IN STUD (1) */        VAN02640
CALL STUD(1) ; N=N+1 ; END ;                                               VAN02650
END TESTY ;                                                                VAN02660
                                                                           VAN02670
/*************************************************************/           VAN02680
/* ORDERING OF THE EQUATIONS FOLLOWING 'VAN DER GIESSEN METHOD' */         VAN02690
/*************************************************************/           VAN02700
/*-------------*/                                                         VAN02710
    RNG:PROC;                                                              VAN02720
/*-------------*/                                                         VAN02730
DCL IT BIT(1);                                                             VAN02740
DCL A PIC'999', /* POINTER IN VECTOR 'AUX'  */                             VAN02750
    G PIC'999', /* POINTER IN VECTOR 'GIVE' */                             VAN02760
```

```
    S PIC'999', /* POINTER IN VECTOR 'SEQ'  */                          VAN02770
    K PIC'999', /* POINTER IN THE CURRENT ROW OF 'T' */                  VAN02780
     (I,J) PIC'999' ;                                                    VAN02790
             /*  INITIATION OF VECTOR 'AUX' */                          VAN02800
A=1;                                                                     VAN02810
DO WHILE(A<=N);                                                          VAN02820
G=0;S=1;                                                                 VAN02830
DO WHILE(S<=N);                                                          VAN02840
IF(T(A,S)='1'B) THEN G=G+1;                                              VAN02850
S=S+1;                                                                   VAN02860
END;AUX(A)=G;A=A+1;END;                                                  VAN02870
       /* WE WANT TO SEPARATE THE RECURSIVE OF THE SYSTEM FROM THE */    VAN02880
       /* SIMULTANEOUS PART. IN MATRIX 'T', 'RECURSIVE VARIABLES'  */    VAN02890
       /* HAVE NO '1'B IN THEIR COLUMN.IN THE FOLLOWING LOOP WE    */    VAN02900
       /* LOOK FOR SUCH VARIABLES AND PUT THEM IN VECTOR 'REC'.    */    VAN02910
R1=1; REC(1)=1;                                                          VAN02920
DO A=1 TO N;                                                             VAN02930
G=0;S=1;                                                                 VAN02940
DO WHILE(S<=N);                                                          VAN02950
IF (T(S,A)='1'B) THEN G=G+1;                                             VAN02960
S=S+1;                                                                   VAN02970
END;                                                                     VAN02980
IF G=0 THEN DO;                                                          VAN02990
REC(R1)=A; R1=R1+I ;END;                                                 VAN03000
END;                                                                     VAN03010
I=1;G=1;S=1;                                                             VAN03020
IT='0'B;                                                                 VAN03030
X10:A=1;                                                                 VAN03040
          /* X11 TEST TO FIND '0' IN VECTOR 'AUX' WHICH MEANS THAT   */VAN03050
          /* THE CORRESPONDING VARIABLE MUST BE PUT IN 'SEQ'         */VAN03060
X11:IF (AUX(A)=0)THEN GOTO X20;                                          VAN03070
   A=A+1;                                                                VAN03080
   IF (A<=N)THEN GOTO X11;                                               VAN03090
        /* WE WENT TO SEPARATE VARIABLES INDEPENDANT FROM ENDOGENOUS */VAN03100
        /* VARIABLES FROM THE OTHER VARIABLES. WE KNOW THAT THESE    */VAN03110
        /* VARIABLES ARE PUT IN 'SEQ' AT THE BEGINNING OF THE ALGO   */VAN03120
        /* RITHM(IT='0'B).SO WE CAN EASILY FIND THIS SEQUENCE OF     */VAN03130
        /* VARIABLES AND THEN WRITE A COMMENT OF OUR CHOICE IN THE   */VAN03140
        /* RIGHT PLACE OF THE OUTPUT FILE                            */VAN03150
   IF (IT='0'B) THEN                                                     VAN03160
    DO;                                                                  VAN03170
    IF (S>1) THEN                                                        VAN03180
     DO;                                                                 VAN03190
     ALLOCATE TAB SET(Q);                                                VAN03200
     COUR1->REL=Q; Q->REL=ENTER; Q->NUM=SEQ(S-1); COUR1=Q;               VAN03210
     ALLOCATE TEXT SET (P);                                              VAN03220
     COUR2=P;                                                            VAN03230
     Q->VERS=P ;                                                         VAN03240
     P->INFO ='C****      END OF RECURSIVE PART      *****';             VAN03250
     P->FOL=ENTER;                                                       VAN03260
     END;                                                                VAN03270
    END;                                                                 VAN03280
   IT='1'B;                                                              VAN03290
         /* X30 RESEARCH OF THE FIRST 1 IN VECTOR 'AUX'  */              VAN03300
X30:A=1;                                                                 VAN03310
X31:IF (AUX(A)=1) THEN GOTO  X40;                                        VAN03320
    A=A+1;                                                               VAN03330
   IF (A<=N) THEN GOTO X31;                                              VAN03340
X50:I=I+1;                                                               VAN03350
    GOTO X25;                                                            VAN03360
         /* X40 RESEARCH OF THE FIRST 1 IN ROW 'A' OF THE MATRIX 'T' */VAN03370
X40:K=1;                                                                 VAN03380
X41:IF (T(A,K)='1'B) THEN GOTO X60;                                      VAN03390
   K=K+1;                                                                VAN03400
    IF (K<=N) THEN GOTO X41;                                             VAN03410
     /* WE HAVE FOUND A '1'B IN COLUMN K ROW A.  X60 WE ADD COL'A' TO*/VAN03420
     /* COL'K' AND ALSO MODIFY 'AUX' AFTER THIS OPERATION           */VAN03430
X60:       DO B=1 TO N;                                                  VAN03440
   IF((T(B,K)='0'B)&(T(B,A)='1'B)) THEN                                  VAN03450
   DO;                                                                   VAN03460
```

```
     AUX(B)=AUX(B)+1;                                                    VAN03470
     T(B,K)='1'B;                                                        VAN03480
     END;                                                                VAN03490
    END;                                                                 VAN03500
        /* RESEARCH OF A '1'B IN THE DIAGONAL OF MATRIX 'T'    */        VAN03510
X70:J=1;                                                                 VAN03520
X71:IF (T(J,J)='1'B) THEN GOTO X80;                                      VAN03530
    J=J+1;                                                               VAN03540
    IF (J<=N) THEN GOTO X71;                                             VAN03550
    IF(I=1) THEN GOTO X35;                                               VAN03560
        /* WE HAVE ALREADY FOUND ONE OR MORE '1'B IN ROW A. THEIR   */   VAN03570
        /* TREATMENT HAS BEEN DONE. WE RESEARCH THE FOLLOWING '1'B. */   VAN03580
X45:K=K+1;                                                               VAN03590
X46:IF(T(A,K)='1'B) THEN GOTO X60;                                       VAN03600
  K=K+1;                                                                 VAN03610
  IF(K<=N) THEN GOTO X46;                                                VAN03620
        /* WE HAVE ALREADY TREATED ONE OR MORE '1' IN VECTOR    */       VAN03630
        /* AUX. WE  RESEARCH THE FOLLOWING '1'                  */       VAN03640
X35:A=A+1;                                                               VAN03650
X36:IF (AUX(A)=1) THEN GOTO X40;                                         VAN03660
    A=A+1;                                                               VAN03670
    IF(A<=N) THEN GOTO X36;                                              VAN03680
     GOTO X50;                                                           VAN03690
       /* WE HAVE FOUND A VARIABLE TO BE GIVEN.  SO WE DELETE THE      */VAN03700
       /* CORRESPONDING COLUMN J,  WE PUT THIS VARIABLE IN 'GIVE'      */VAN03710
       /* AND ADJUST 'AUX'                                             */VAN03720
X80:A=1;                                                                 VAN03730
X81:IF(T(A,J)='0'B) THEN GOTO X82;                                       VAN03740
    T(A,J)='0'B;                                                         VAN03750
     AUX(A)=AUX(A)-1;                                                    VAN03760
X82:A=A+1;                                                               VAN03770
  IF(A<=N) THEN GOTO X81;                                                VAN03780
    GIVE(G)=J;                                                           VAN03790
     G=G+1;                                                              VAN03800
     IF(I=1) THEN GOTO X10;                                              VAN03810
    I=I-1;                                                               VAN03820
     GOTO X30 ;                                                          VAN03830
      /* WE HAVE FOUND A ROW 'A' WITH AUX(A)=0.  WE PUT THIS VARIABLE */VAN03840
      /* IN VECTOR 'SEQ', WE DELETE THE CORRESPONDING COLUMN IN       */VAN03850
      /* MATRIX 'T' AND ADJUST 'AUX'                                  */VAN03860
X20:SEQ(S)=A;                                                            VAN03870
    AUX(A)=900;                                                          VAN03880
    S=S+1;                                                               VAN03890
    J=1;                                                                 VAN03900
X21:IF (T(J,A)='0'B) THEN GOTO X22;                                      VAN03910
     T(J,A)='0'B;                                                        VAN03920
     AUX(J)=AUX(J)-1;                                                    VAN03930
X22:J=J+1;                                                               VAN03940
    IF(J<=N) THEN GOTO X21;                                              VAN03950
        /* X25 TEST TO KNOW IF THE SEQUENCE IS FINISHED  */              VAN03960
X25:IF(S=N+1) THEN GOTO FIN ;                                            VAN03970
    ELSE GOTO X10;                                                       VAN03980
FIN:END RNG;                                                             VAN03990
                                                                         VAN04000
/*****************************************************************/     VAN04010
/* THIS SUBROUTINE SEARCHS IF 'IN' IS ONE OF THE COMPONENTS OF   */     VAN04020
/* VECTOR 'REC'. IF YES TROUVE=1  ELSE TROUVE=0                  */     VAN04030
/*****************************************************************/     VAN04040
/*------------------*/                                                  VAN04050
    COMP:PROC(IN);                                                       VAN04060
/*------------------*/                                                  VAN04070
DCL  IN PIC'999',                                                        VAN04080
     R2 PIC'999',                                                        VAN04090
     TROUVE PIC'9';                                                      VAN04100
TROUVE=0; R2=1;                                                          VAN04110
DO WHILE (R2<R1);                                                        VAN04120
IF (REC(R2)=IN) THEN DO; TROUVE=1; R2=R1; END;                           VAN04130
ELSE R2=R2+1;                                                            VAN04140
END;                                                                     VAN04150
RETURN(TROUVE);                                                          VAN04160
```

```
END COMP;                                                                    VAN04170
                                                                             VAN04180
/*************************************************************************/ VAN04190
/* THIS SUBROUTINE WRITES IN THE OUTPUT FILE ALL THE LISTS 'TEXT' */        VAN04200
/* HAVING THE SAME INDEX 'NUM' IN THE LIST 'TAB'.                 */        VAN04210
/*************************************************************************/ VAN04220
/*------------------------*/                                                VAN04230
    IMPRIM:PROC(INDIC);                                                      VAN04240
/*------------------------*/                                                VAN04250
DCL INDIC PIC'999';                                                          VAN04260
RANG1=ENTER->REL;                                                            VAN04270
     /* WE LOOK IF THERE IS  MORE THAN ONE LIST WITH THE         */         VAN04280
     /* SAME 'NUM' AND WE CALL ECRIT AS MANY TIMES AS NECESSARY */         VAN04290
DO WHILE(RANG1^=ENTER);                                                      VAN04300
CALL ECRIT(INDIC,RANG1);                                                     VAN04310
IF (COUR1^=ENTER) THEN RANG1=COUR1->REL;                                     VAN04320
ELSE RANG1=ENTER;                                                            VAN04330
END;                                                                         VAN04340
END IMPRIM;                                                                  VAN04350
                                                                             VAN04360
              /*************************************/                        VAN04370
              /****                             ****/                        VAN04380
              /*            MAIN PROCEDURE         */                        VAN04390
              /****                             ****/                        VAN04400
              /*************************************/                        VAN04410
                                                                             VAN04420
    /* INITIATION */                                                         VAN04430
ALLOCATE TAB SET(Q);                                                         VAN04440
ENTER=Q;                                                                     VAN04450
COUR1=Q;                                                                     VAN04460
Q->REL=ENTER;                                                                VAN04470
T='0'B;N=0;                                                                  VAN04480
COM='0'B;                                                                    VAN04490
OPEN FILE(ONE) INPUT;                                                        VAN04500
OPEN FILE(TWO) OUTPUT;                                                       VAN04510
READ FILE(ONE) INTO (CARD);                                                  VAN04520
     /* LOOP TO FIND THE EQUATIONS, STORE THEM AND FILL MATRIX 'T' */       VAN04530
ON ENDFILE(ONE) GOTO SUIT ;                                                  VAN04540
  LOOP:IF (SUBSTR(CARD,1,1)='C') THEN                                        VAN04550
    DO;                                                                      VAN04560
          /* WE ARE ON A COMMENT CARD.COM='0'B MEANS THAT */                 VAN04570
          /* THIS IS THE FIRST COMMENT CARD COMING AFTER  */                 VAN04580
          /* AN EQUATION OR IF STATEMENT SO WE CREATE     */                 VAN04590
          /* A NEW LIST TEXT AND BEGIN IT WITH THIS CARD  */                 VAN04600
    IF (COM='0'B) THEN                                                       VAN04610
      DO;                                                                    VAN04620
      ALLOCATE TAB SET(Q);                                                   VAN04630
      COUR1->REL=Q; Q->REL=ENTER;COUR1=Q; COM='1'B;                          VAN04640
      ALLOCATE TEXT SET(P);                                                  VAN04650
      COUR2=P; Q->VERS=P; P->INFO=CARD; P->FOL=ENTER;                        VAN04660
      END;                                                                   VAN04670
    ELSE                                                                     VAN04680
          /* WE CONTINUE THE 'COMMENT LIST ' ALREADY BEGUN   */              VAN04690
      DO;                                                                    VAN04700
      ALLOCATE TEXT SET(P);                                                  VAN04710
      COUR2->FOL=P;                                                          VAN04720
      P->INFO=CARD;                                                          VAN04730
      P->FOL=ENTER;                                                          VAN04740
      COUR2=P;                                                               VAN04750
      END;                                                                   VAN04760
  READ FILE(ONE) INTO (CARD) ;                                               VAN04770
    END;                                                                     VAN04780
  ELSE                                                                       VAN04790
    /* WE ARE NOT ON A COMMENT SO WE HAVE TO DO THE WHOLE TREATMENT */      VAN04800
    DO;                                                                      VAN04810
    CALL CREATE;                                                             VAN04820
  CALL TESTY;                                                                VAN04830
    CALL ST;                                                                 VAN04840
    COM='0'B;                                                                VAN04850
    END;                                                                     VAN04860
```

```
GOTO LOOP;                                                              VAN04870
          /*  ORDERING OF EQUATIONS   */                                VAN04880
SUIT:CALL RNG;                                                          VAN04890
        /* WE DO SEQ(N+1)=999 TO PRINT IF STATEMENT WITHOUT  */         VAN04900
        /* EQUATION Y(..)=..   AFTER TEST IF                 */         VAN04910
   SEQ(N+1)=999;                                                        VAN04920
        /* PRINTING OF VECTORS 'GIVE' AND 'SEQ'   */                    VAN04930
PUT SKIP  LIST('GIVE','SEQUENCE');                                      VAN04940
PUT SKIP LIST(' ' );                                                    VAN04950
DO I=1 TO N;                                                            VAN04960
PUT SKIP LIST(GIVE(I),SEQ(I));                                          VAN04970
END;                                                                    VAN04980
     /* PRINTING OF THE EQUATIONS WITH NEW ORDERING */                  VAN04990
DO I=1 TO N+1;                                                          VAN05000
        /* FIRST WE LOOK IF THE VARIABLE(SEQ(I))  BELONGS TO THE */     VAN05010
        /* 'RECURSIVE PART' OF THE SYSTEM.IF YES(COM(SEQ(I))=1   */     VAN05020
        /* IT WILL BE WRITTEN LATER,OTHERWISE(COM(SEQ(I))=0) IT  */     VAN05030
        /* IS WRITTEN AT ONCE.                                   */     VAN05040
IF (COMP(SEQ(I))=0) THEN                                                VAN05050
CALL IMPRIM(SEQ(I));                                                    VAN05060
END;                                                                    VAN05070
        /* R1>1  MEANS THAT THERE ARE SOME EQUATIONS IN 'RECURSIVE  */ VAN05080
        /* PART' OF THE SYSTEM. THESE EQUATIONS ARE IN VECTOR 'REC' */ VAN05090
        /* WE PUT THEM IN THE OUTPUT FILE 'TWO'                     */ VAN05100
IF (R1>1) THEN                                                          VAN05110
DO;                                                                     VAN05120
  CARD=' ';                                                             VAN05130
  CARD='C**  END OF SIMULTANEOUS PART  ***';                            VAN05140
  WRITE FILE(TWO) FROM (CARD);                                          VAN05150
  DO I=1 TO R1-1;                                                       VAN05160
   CALL IMPRIM(REC(I));                                                 VAN05170
  END;                                                                  VAN05180
END;                                                                    VAN05190
CLOSE FILE(ONE);                                                        VAN05200
CLOSE FILE(TWO);                                                        VAN05210
END VANDERG;                                                            VAN05220
```

## REFERENCES

[1] Bianchi,C., G.Calzolari and P.Corsi, "A Program for Stochastic Simulation of Econometric Models", Econometrica, 46 (1978), 235-236.

[2] Bianchi,C., G.Calzolari and P.Corsi, "Stochastic Simulation of Econometric Models: Installation Procedures and User's Instructions", IBM Technical Report G513-3568, Pisa, (1978).

[3] IBM, "Virtual Machine Facility/370: CMS Command and Macro Reference", GC20-1818, IBM, New York, (1976).

[4] Klein,L.R., "Estimation of Interdependent Systems in Macroeconometrics", Econometrica, 37 (1969), 171-192.

[5] Van der Giessen,A.A., "Solving Non-Linear Systems by Computer; a New Method", Statistica Neederlandica, 24 (1970), 41-50.