INVCH - A FORTRAN PACKAGE FOR CALCULATING THE INVERSE OF A PERTURBED MATRIX

J.W. Bandler and Q.J. Zhang

SOS-84-22-U

December 1984

INVCH – A FORTRAN PACKAGE FOR CALCULATING THE INVERSE OF A PERTURBED MATRIX

J.W. Bandler and Q.J. Zhang

Abstract

INVCH is a package of subroutines for calculating the inverse of a perturbed matrix. The package uses an iterative procedure to calculate large change effects. The user should supply the perturbation matrix in a product form. The method is described in detail by our previous report. The package and documentation have been developed for the CDC 170/815 system with the NOS 2.2-602/587 operating system and the Fortran Extended (FTN) version 4.8 compiler.

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under Grant A7239.

The authors are with the Simulation Optimization System Research Laboratory and the Department of Electrical and Computer Engineering, McMaster University, Hamilton, Canada L8S 4L7.

I. INTRODUCTION

The process of calculating the inverse of a perturbed square matrix can be essentially reduced to the computation of a relatively smaller system. In our investigation of large change effects in linear systems, we described an iterative process which uses a rank 1 change formula to calculate perturbed matrix inversions [1]. INVCH is a package implementing this method. The user should supply the large changes of the matrix in the form of $\mathbf{VDW}^T$, where $\mathbf{V}$, $\mathbf{D}$ and $\mathbf{W}$ are matrices of dimensions $n \times r_1$, $r_1 \times r_2$, and $n \times r_2$, respectively.

There are two entries to the package. The user can call from either Entry 1 or Entry 2.

Calling from Entry 1, the user will go through an interactive procedure with the computer for easy and direct access to large change computations. The calling procedure is very simple. Entry 1 is recommended when the package is used directly for the inversion of a perturbed matrix as final output results and when the numerical perturbations can be practically entered from the computer terminal.

Calling from Entry 2, the user will use the basic subroutines for large change calculations. The package can be used at this entry as a subsection of a more comprehensive program which requires matrix inversions only as intermediate steps. Also, at this entry, the perturbations of the linear system can be supplied from, e.g., variable updating calculations, not necessarily from a computer terminal.

The whole package is written in Fortran IV for the CDC-170/815 system. At McMaster University, it is available in the form of a library of binary relocatable subroutines which is linked with the user's program by an appropriate call to the package. The name of the library is LIBILCH. The library is available as a group indirect file under the charge RJWBAND. The general sequence of NOS commands to use the package can be as follows:

/GET (LIBILCH/GR) - fetch the library,

/LIBRARY (LIBILCH) - indicate the library to the loader.

The user's program should be composed (at least) of the main segment which prepares parameters and calls the subroutines of the package corresponding to either Entry 1 or Entry 2.

This document includes the user's manual of the INVCH package together with numerical examples. A Fortran listing of the package is found in [2].

II. THEORETICAL REVIEW

Formulation of the Problem

Let $\mathbf{A}$ be an n x n matrix containing variables. Suppose the values of the variables are changed causing changes of matrix $\mathbf{A}$ such that

$$\mathbf{A}_{\text{new}} = \mathbf{A} + \Delta\mathbf{A}\,. \tag{1}$$

If we use direct methods, the inversion of $\mathbf{A}_{\text{new}}$ can be obtained by applying matrix inversion techniques on the n x n matrix $\mathbf{A}_{\text{new}}$. However, when changes in the system only affect a submatrix of $\mathbf{A}$, a more elegant approach for $\mathbf{A}_{\text{new}}{}^{-1}$ is to use large change formulas which can essentially reduce the computation to a relatively smaller system depending upon the rank of $\Delta\mathbf{A}$.

In our previous report [1], we introduced a series of formulas for large change calculations, among which, formula (64) is used in the INVCH package.

Large Change Formula (64)

Formula (64) of [1] gives an iterative procedure which uses the rank 1 change formula for the calculation of large change effects with the rank of $\Delta\mathbf{A}$ higher than 1, i.e.,

$$\mathbf{A}_{\text{new}}{}^{-1} = \mathbf{R}_{r_2} \tag{2}$$

where

$$\mathbf{R}_0 = \mathbf{A}^{-1} \tag{3}$$

and

$$\mathbf{s}_k = \mathbf{R}_{k-1}\,\mathbf{V}\,\mathbf{D}\,\mathbf{u}_k \tag{Formula 64a}$$

$$\mathbf{w}_k = \mathbf{W}\,\mathbf{u}_k \tag{Formula 64b}$$

$$\mathbf{R}_k = \mathbf{R}_{k-1} - \frac{\mathbf{s}_k\,\mathbf{w}_k^T\,\mathbf{R}_{k-1}}{1 + \mathbf{w}_k^T\,\mathbf{s}_k} \tag{Formula 64c}$$

$$k = 1, 2, \ldots, r_2$$

where **V**, **D**, and **W** are matrices of dimensions $n \times r_1$, $r_1 \times r_2$ and $n \times r_2$ respectively such that

$$\Delta\mathbf{A} = \mathbf{V}\,\mathbf{D}\,\mathbf{W}^T . \tag{4}$$

Discussion

The approach of Formula (64) is preferred especially if **V**, **D** and **W** have been formulated with $r_1 \geq r_2$, otherwise, the dual properties [1] can be used to yield a similar iterative procedure.

We can always formulate **V**, **D** and **W** such that $r_1 \leq n$ and $r_2 \leq n$. The smaller r_1 and r_2, the better. The minimal value of r_1 or r_2 is

$$\min_{(\mathbf{V},\mathbf{D},\mathbf{W})} r_1 = \min_{(\mathbf{V},\mathbf{D},\mathbf{W})} r_2 = r , \tag{5}$$

where r is the rank of $\Delta\mathbf{A}$. Detailed information on the formulation of **V**, **D** and **W** can be found in [1].

III. STRUCTURE OF THE PACKAGE

Figures 1 and 2 show possible structures of the INVCH package corresponding to Entries 1 and 2, respectively.

Figure 1 shows the package structure yielding a user-computer interactive procedure. Subroutines INVCH and INVER0 are designed to organize the interactive procedure and guide the user through the user-computer dialogue. INVCHA is the basic subroutine which implements large change Formula (64). LUFACT is a subroutine of LU factorization and/or forward and backward substitutions (FBS) which is used to perform the original matrix inversion. The main program must be supplied by the user. In the main program, the user only needs to call INVCH. r_1, r_2, **V**, **D** and **W** are entered from the computer terminal through an interactive procedure guided by the package.

Figure 2 shows the package structure for Entry 2. Only the basic subroutine, i.e. INVCHA, is used. The main program must be supplied by the user which may be a more comprehensive program requiring matrix inversions only as intermediate steps.

IV. ARGUMENT LISTS

For Entry 1

The subroutine call is

```
CALL   INVCH (N, LW0, A, R, W0, ICH)
```

The arguments are as follows.

N	is an INTEGER argument which must be set to n, the number of rows or columns of **A**. Its value must be positive and it is not altered by the package.
LW0	is an INTEGER argument which must be set to the length of workspace W0. Its value must be at least $$n(2 + n + r_1 + r_2) + r_1 r_2$$ where r_1 and r_2 are the number of rows and columns of **D**, respectively. LW0 = n (2 + 4n) may be used to provide easy evaluation and sufficient value for LW0.
A	is a REAL array of dimensions (N,N). On entry, it must be set to **A**, the original matrix. On exit, it contains the LU factors of **A**.
R	is a REAL array of dimensions (N,N). On exit, it contains $\mathbf{A}^{-1}$, the original matrix inversion, i.e. before large change.
W0	is a REAL array used as workspace. Its length is given by LW0.
ICH	is an INTEGER argument which must be set to the unit number (or channel number) that is to be used for the printed output generated by the package. Usually, it is the unit number of the file OUTPUT.

<u>For Entry 2</u>

The subroutine call is

CALL INVCHA (N, IR1, IR2, R, V, D, W, S, G, IFLAG).

The arguments are as follows.

N	is an INTEGER argument which must be set to n, the number of rows or columns of **A**. Its value must be positive and it is not altered by the package.
IR1	is an INTEGER argument which must be set to r_1, the number of rows of **D**. Its value must be positive and it is not altered by the package.
IR2	is an INTEGER argument which must be set to r_2, the number of columns of **D**. Its value must be positive and it is not altered by the package.
R	is a REAL array of dimensions (N,N). On entry, it must be set to the original matrix inverse, i.e. the inverse of **A** before perturbation. On exit, it contains the inverse of $(\mathbf{A} + \mathbf{VDW}^T)$, i.e. the inverse of the perturbed matrix.
V	is a REAL array of dimensions (N, IR1). On entry, it must be set to **V** (see the definition of **V** in (4)) and it is not altered by the subroutine.
D	is a REAL array of dimensions (IR1, IR2). On entry, it must be set to **D** (see the definition of **D** in (4)) and it is not altered by the subroutine.
W	is a REAL array of dimensions (N, IR2). On entry, it must be set to **W** (see the definition of **W** in (4)) and it is not altered by the subroutine.
S	is a REAL array used as workspace. Its length is given by N.
G	is a REAL array used as workspace. Its length is given by N.
IFLAG	is an INTEGER argument which, on exit, contains information about the result:

IFLAG = 0	successful exit.
IFLAG = 1	calculation interrupted because zero occurs in the denominator of (Formula 64c).

V. EXAMPLES

Example 1 (The INVCH package is called at Entry 1)

Consider the 4 x 4 matrix,

$$\mathbf{A} = \begin{bmatrix} 1 & 4 & 2 & 4 \\ 2 & 3 & 0 & 8 \\ 3 & 2 & 9 & 1 \\ 4 & 1 & 5 & 9 \end{bmatrix} . \tag{6}$$

Various perturbations to the matrix are expressed by different sets of **V**, **D** and **W**. Together with r_1 and r_2, the number of rows and columns of **D** respectively, **V**, **D** and **W** are entered through an interactive procedure. The following shows the main program, which is supplied by the user, and the user-computer dialogue procedure.

```
      PROGRAM EXAMP1(INPUT,OUTPUT,TAPE6=OUTPUT)
C
C     EXAMPLE 1.
C
      DIMENSION A(4,4),R(4,4),W0(100)
C
      DATA A/1.,2.,3.,4.,
     +       4.,3.,2.,1.,
     +       2.,0.,9.,5.,
     +       4.,8.,1.,9./
      DATA N,LW0,ICH/4,100,6/
C
      CALL INVCH(N,LW0,A,R,W0,ICH)
      STOP
      END
```

```
    THE ORIGINAL MATRIX [A] :

        1.00000      4.00000      2.00000      4.00000

        2.00000      3.00000      0.00000      8.00000

        3.00000      2.00000      9.00000      1.00000

        4.00000      1.00000      5.00000      9.00000


***********************************************************
 FIND THE INVERSE OF ( [A]+[V][D][W]' ) BY LARGE CHANGE
 FORMULA .

     THE INVERSE OF [A] BEFORE CHANGE :

       -2.14545      2.49091      1.25455     -1.40000

        -.14545       .49091       .25455      -.40000

         .68182      -.86364      -.31818       .50000

         .59091      -.68182      -.40909       .50000


************************************************************
 ENTER IR1 AND IR2    ( SUCH THAT [D] IS OF IR1 BY IR2 )
*INPUT*  3,2
 [V],[D] AND [W] ARE MATRICES OF DIMENSIONS N BY IR1,
                 IR1 BY IR2 AND N BY IR2 RESPECTIVELY
                 SUCH THAT  DELTA([A])=[V][D][W]' .
 ENTER [V]   ( COLUMN BY COLUMN )
*INPUT*  1.,0.,0.,0.
*INPUT*  0.,1.,0.,0.
*INPUT*  0.,0.,0.,1.
 ENTER [W]   ( COLUMN BY COLUMN )
*INPUT*  0.,1.,0.,0.
*INPUT*  0.,0.,1.,0.
    NUMBER OF ROWS OR COLUMNS OF [A] ( N )    .........  4

    NUMBER OF ROWS OF [D] ( IR1 )             .........  3

    NUMBER OF COLUMNS OF [D] ( IR2 )          .........  2


     MATRIX [V] :

        1.00000      0.00000      0.00000

        0.00000      1.00000      0.00000

        0.00000      0.00000      0.00000

        0.00000      0.00000      1.00000
```

```
      MATRIX [W] :

          0.00000      0.00000

          1.00000      0.00000

          0.00000      1.00000

          0.00000      0.00000


 [D] IS OF 3 BY 2.
 ENTER [D]  ( COLUMN BY COLUMN )
*INPUT*  1.,2.,3.
*INPUT*  4.,5.,6.

      MATRIX [D] :

          1.00000      4.00000

          2.00000      5.00000

          3.00000      6.00000


      THE INVERSE OF ( [A]+[V][D][W]' ) :

          -.86154      1.07692       .78462      -.66154

           .00513       .41026       .25128      -.39487

           .28205      -.43590      -.17949       .28205

           .03590      -.12821      -.24103       .23590


 TRY ANOTHER [D] ?     ENTER 1 OR 2  ( Y OR N )
*INPUT*  1
 [D] IS OF 3 BY 2.
 ENTER [D]  ( COLUMN BY COLUMN )
*INPUT*  3.,4.,5.
*INPUT*  0.,8.,3.

      [V] AND [W] ARE THE SAME AS THOSE WITH THE PREVIOUS [D].

      MATRIX [D] :

          3.00000      0.00000

          4.00000      8.00000

          5.00000      3.00000


      THE INVERSE OF ( [A]+[V][D][W]' ) :

           .19039      -.57461       .09949       .41509

           .22985      -.03602       .04803      -.07547

          -.09777       .18696       .08405      -.13208

          -.15094       .11321      -.15094       .09434
 TRY ANOTHER [D] ?     ENTER 1 OR 2  ( Y OR N )
*INPUT*  2
 TRY ANOTHER SET OF  [V], [D] AND [W] ?
 ENTER 1 OR 2  ( Y OR N )
*INPUT*  1
*************************************************************
 ENTER IR1 AND IR2     ( SUCH THAT [D] IS OF IR1 BY IR2 )
*INPUT*  1,2
 [V],[D] AND [W] ARE MATRICES OF DIMENSIONS N BY IR1,
                 IR1 BY IR2 AND N BY IR2 RESPECTIVELY
                 SUCH THAT  DELTA([A])=[V][D][W]' .
 ENTER [V]   ( COLUMN BY COLUMN )
*INPUT*  1.,-2.,4.,7.
 ENTER [W]   ( COLUMN BY COLUMN )
*INPUT*  2.,0.,4.,5.
*INPUT*  3.,1.,1.,2.
     NUMBER OF ROWS OR COLUMNS OF [A] ( N )    .........   4

     NUMBER OF ROWS OF [D] ( IR1 )              .........   1

     NUMBER OF COLUMNS OF [D] ( IR2 )           .........   2


      MATRIX [V] :

          1.00000

         -2.00000

          4.00000

          7.00000


      MATRIX [W] :

          2.00000      3.00000

          0.00000      1.00000

          4.00000      1.00000

          5.00000      2.00000


 [D] IS OF 1 BY 2.
 ENTER [D]   ( COLUMN BY COLUMN )
*INPUT*  1.,3.

      MATRIX [D] :

          1.00000      3.00000


      THE INVERSE OF ( [A]+[V][D][W]' ) :

           .21952     -.34908     -.19745      .00820

           .43225     -.20283     -.10014     -.05601

          -.23890      .24201      .24710     -.04823

          -.16733      .22871      .05644      .04851
 TRY ANOTHER [D] ?    ENTER 1 OR 2  ( Y OR N )
*INPUT*  1
 [D] IS OF 1 BY 2.
 ENTER [D]  ( COLUMN BY COLUMN )
*INPUT*  3.,7.

      [V] AND [W] ARE THE SAME AS THOSE WITH THE PREVIOUS [D].


     MATRIX [D] :

         3.00000     7.00000


     THE INVERSE OF ( [A]+[V][D][W]' ) :

           .27294      -.40141      -.23276      -.00870

           .44530      -.21561      -.10877      -.06014

         -.25969       .26238       .26084      -.04165

         -.18445       .24549       .06775       .05393


 TRY ANOTHER [D] ?    ENTER 1 OR 2  ( Y OR N )
*INPUT*  2
 TRY ANOTHER SET OF  [V], [D] AND [W] ?
 ENTER 1 OR 2  ( Y OR N )
*INPUT*  2
```

<u>Example 2</u> (The INVCH package is called at Entry 2)

Consider the 5 x 5 matrix,

$$\mathbf{A} = \begin{bmatrix} 2 & 4 & 3 & 3 & 4 \\ 1 & 6 & 9 & 6 & 0 \\ 5 & 7 & 2 & 5 & 9 \\ 0 & 2 & 1 & 4 & 3 \\ 9 & 1 & 0 & 1 & 6 \end{bmatrix} . \tag{7}$$

The variable parameters of the system are A_{12}, A_{14}, A_{32}, A_{34}, A_{42} and A_{44}. An easy expression of $\Delta\mathbf{A}$ in the form of $\mathbf{VDW}^T$ can be such that

$$\mathbf{V} = [\mathbf{u}_1\ \mathbf{u}_3\ \mathbf{u}_4], \tag{8}$$

$$\mathbf{W} = [\mathbf{u}_2\ \mathbf{u}_4] \tag{9}$$

and

$$\mathbf{D} = \begin{bmatrix} \Delta A_{12} & \Delta A_{14} \\ \Delta A_{32} & \Delta A_{34} \\ \Delta A_{42} & \Delta A_{44} \end{bmatrix} , \tag{10}$$

where $\mathbf{u}_i$, i = 1, 2, 3, 4, are unit 5-vectors with 1 at the ith row and zeros everywhere else. Suppose the variable parameters are perturbed 4 times according to Table I. The following shows the main program and the output results.

```
      PROGRAM EXAMP2(INPUT,OUTPUT,TAPE6=OUTPUT)
C
C     EXAMPLE 2.
C
      DIMENSION A(5,5),V(5,3),D(3,2),W(5,2),DD(3,2,4)
      DIMENSION R(5,5),S(5),G(5)
C
      DATA N,IR1,IR2/5,3,2/
      DATA A/2.,1.,5.,0.,9.,
     +       4.,6.,7.,2.,1.,
     +       3.,9.,2.,1.,0.,
     +       3.,6.,5.,4.,1.,
     +       4.,0.,9.,3.,6./
      DATA V,W/1.,0.,0.,0.,0.,    0.,0.,1.,0.,0.,
     +          0.,0.,0.,1.,0.,
     +          0.,1.,0.,0.,0.,    0.,0.,0.,1.,0./
      DATA DD/2.,3.,4., 5.,3.,8.,     4.,6.,9., 1.,2.,4.,
     +        3.,4.,7., 1.,2.,9.,     2.,4.,9., 0.,4.,1./
C
C     CALCULATE THE INVERSE OF THE ORIGINAL MATRIX.
C
      DO 20 J=1,N
      DO 10 I=1,N
   10 S(I)=0.
      S(J)=1.
C
C     MODE=1 FOR LU-FACTORIZATION AND FBS.,  MODE=3 FOR FBS. ONLY.
C
      MODE=3
      IF(J.EQ.1) MODE=1
      CALL LUFACT(N,A,S,MODE)
      DO 20 I=1,N
   20 R(I,J)=S(I)
C
      PRINT*,"     THE INVERSE OF [A] BEFORE LARGE CHANGE :"
      WRITE(6,40)  ((R(I,J),J=1,N),I=1,N)
C
      DO 30 K=1,4
      DO 25 I=1,IR1
      DO 25 J=1,IR2
   25 D(I,J)=DD(I,J,K)
C
      CALL INVCHA(N,IR1,IR2,R,V,D,W,S,G,IFLAG)
      PRINT*,"     THE INVERSE OF [A] AFTER LARGE CHANGE (",K,")"
      WRITE(6,50) ((R(I,J),J=1,N),I=1,N),IFLAG
   30 CONTINUE
      STOP
   40 FORMAT(/5(10X,5F12.5//)////)
   50 FORMAT(/5(10X,5F12.5//)//10X,"I F L A G  == ",I3////)
      END
```

```
     THE INVERSE OF [A] BEFORE LARGE CHANGE :

               -.50704      .13803      .17746     -.07606      .10986

               -.74648      .16432      .54460     -.32864     -.15493

                .95775     -.10516     -.46854      .01033      .05915

               -.60563      .13709      .12864      .32582      .04789

                .98592     -.25728     -.37840      .11455      .01972
```

```
THE  INVERSE  OF  [A]  AFTER  LARGE  CHANGE  (1)
          -1.27648       .31329       .25045       .50898       .22083
          -1.71275       .38555       .54174       .58483       .03680
           1.13645      -.14811      -.29892      -.47846      -.07002
            .22083      -.04892      -.13510       .04803       .03142
           2.16338      -.52603      -.44345      -.86894      -.17594

     I  F  L  A  G   ==    0


THE  INVERSE  OF  [A]  AFTER  LARGE  CHANGE  (2)
          -1.06940       .26606       .23538       .34287       .18843
          -1.07125       .24053       .39003       .26895      -.00535
            .38859       .02059      -.09246      -.16619      -.03727
            .66660      -.14909      -.29057      -.07681       .02986
           1.67154      -.41433      -.36965      -.54633      -.12006

     I  F  L  A  G   ==    0


THE  INVERSE  OF  [A]  AFTER  LARGE  CHANGE  (3)
          15.75000     -3.50000     -6.75000     -2.25000       .75000
          11.95588     -2.67647     -5.01471     -1.75000       .42647
          -4.73476      1.17914      2.04144       .65909      -.20187
          -7.40374      1.65775      3.07754      1.13636      -.24866
         -24.38369      5.41979     10.44786      3.47727      -.98797

     I  F  L  A  G   ==    0


THE  INVERSE  OF  [A]  AFTER  LARGE  CHANGE  (4)
            .85956      -.16585      -.56621       .04638       .25309
           1.00875      -.22516      -.47753      -.04470       .06615
           -.06651       .12283       .06890      -.04374      -.03715
          -1.05225       .23523       .46854       .10258      -.05261
          -1.28209       .24709       .85081      -.07921      -.21522

     I  F  L  A  G   ==    0
```

REFERENCES

[1] J.W. Bandler and Q.J. Zhang, "A unified approach to first-order and large change sensitivity computations in linear systems", Department of Electrical and Computer Engineering, McMaster University, Hamilton, Canada, Report SOS-84-20-R, 1984.

[2] J.W. Bandler and Q.J. Zhang, "INVCH - A Fortran package for calculating the inverse of a perturbed matrix", Department of Electrical and Computer Engineering, McMaster University, Hamilton, Canada, Report SOS-84-22-L, 1984.

TABLE I

DIFFERENT PERTURBATIONS TO VARIABLES IN EXAMPLE 2

Large Change	ΔA_{12}	ΔA_{14}	ΔA_{32}	ΔA_{34}	ΔA_{42}	ΔA_{44}
1	2	5	3	3	4	8
2	4	1	6	2	9	4
3	3	1	4	2	7	9
4	2	0	4	4	9	1

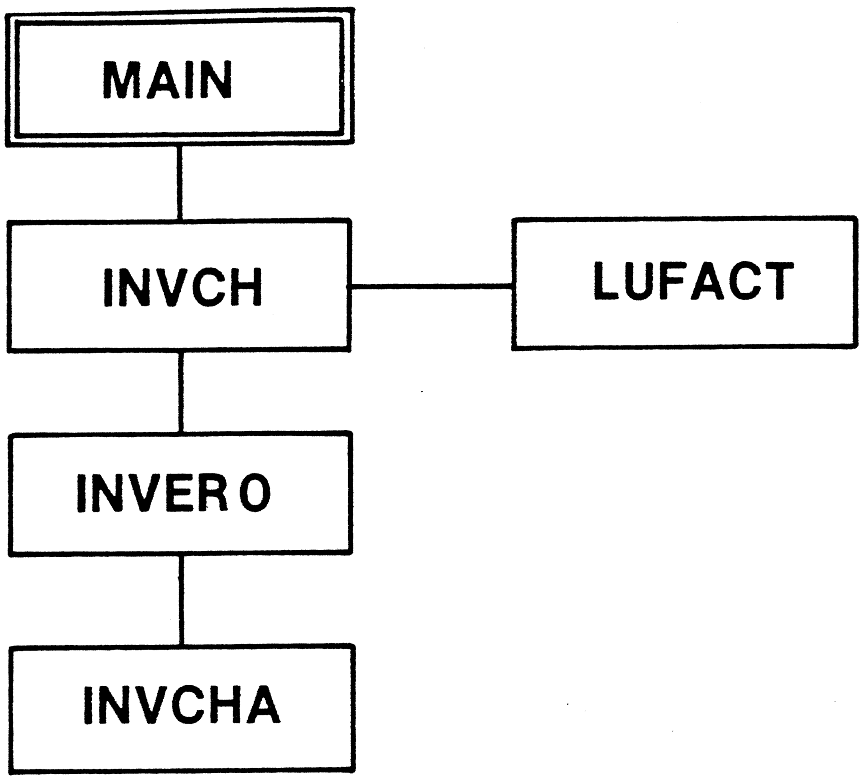

Fig. 1 Structure of the INVCH package corresponding to Entry 1.

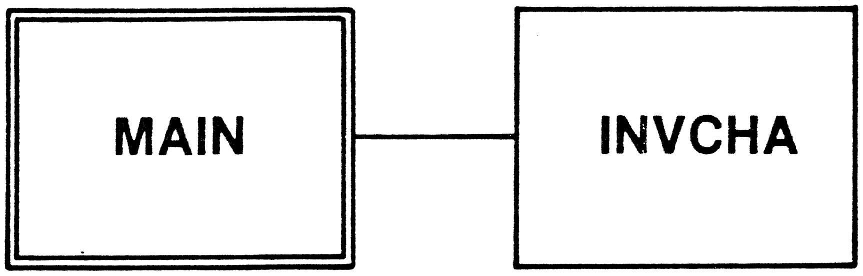

Fig. 2 Structure of the INVCH package corresponding to Entry 2.