tanulmányok

115/1980

MTA Számítástechnikai és Automatizálási Kutató Intézet Budapest

MAGYAR TUDOMÁNYOS AKADÉMIA
SZÁMITÁSTECHNIKAI ÉS AUTOMATIZÁLÁSI KUTATÓ INTÉZETE

PROGRAM PACKAGE SPARSE MATRICES

Irta:

Gergely József

Tanulmányok 115/1980.

A kiadásért felelős:

DR VÁMOS TIBOR

ISBN 963 311 112 9
ISSN 0324 - 2951

Készült a SZÁMOK KSH nyomdájában
226/1980.

Abstract

In the paper programs are investigated for the solution of six problems of linear algebra where the matrices of the problems are sparse ones. Linear systems of equations are solved in the first four programs, the eigenvalues are computed in the 5th and 6th program in the symmetric, non-symmetric or band cases. The inverse and determinant also are computed in the first program.

1. Introduction

A lot of methods and programs for the solution of the problems of linear algebra are known. In the books dealing with linear algebra or numerical methods there are sections in which the methods for the solution of systems of linear equations and matrix inversion are investigated. Moreover, the libraries of computer centers contain various programs to solve problems of linear algebra.

Large size main memories are needed in computers for the solution of large size problems. The solutions of large size problems are computed by means of back store, if the main memory is not enough for the solution. In this case the programs of solutions are more sophisticated and the computation takes more time because of the transfer between the memories, (see [1]).

Computer manipulation of large size matrices is often possible, because the matrix has a number of zero elements such that it is sparse. The subject of the paper is to make known some methods and some programs for the solution of linear equations with sparse matrices, for the inversion of sparse matrices, for the computation of determinant and eigenvalues of sparse matrices.

The main memory can prove to be small in case of a not very large computer even if we utilise possibilities of sparsity of large size matrices. Our programs /except the first one/ are made for small computers. The second, third and fourth programs during the computation use the back store /disc/ also to store the matrices, they have, however such an organization that the transfer times between the memories are minimal.

Some reviews about programs to handle sparse problems can be found in the libraries (see [3], [4], [8],). Generally these programs can be used only in large computers or there is possible to solve only small problems for being used in smaller computers.

Our programs are made in FORTRAN in the computer CDC 3300. We give the memory need at each program.

2. Problems and methods

The problems and the methods for their solution are investigated in this chapter. There are given the sparse matrix $A = \{a_{ij}\}$, $i,j = 1,\ldots,n$ and the vector $b = \{b_i\}$, $i = 1,\ldots,n$. The problems are to solve the system of linear equation $Ax = b$, to invert the matrix A and to compute the eigenvalues of the matrix A and $\det(A)$.

Problem 1 is to solve the system of linear equation $Ax = b$ or to compute the inverse A^{-1} and $\det(A)$.

The Crout method is used for solution. This is the following: the matrix A is factorized $A = LR$ by means of Gauss elimination, where L is a lower triangular matrix, R is an upper triangular matrix. The equation after the factorization is

$$(2.1) \qquad Ax = LRx = Ly = b.$$

The solution of the equation (2.1) is in two steps: the successive solutions of the following two equations

$$(2.2) \qquad Ly = b, \qquad Rx = y.$$

To solve the equations (2.2) is very simple and quick because it can be done by means of substitutions (L and R are triangular).

The equation $AX = I$ is solved for inversion by means of the Crout method, where I is the unit matrix. After factorization $A = LR$ the equation

$$AX = LRX = LY = I$$

is solved in two steps, we have to solve the equations

$$(2.3) \qquad LY = I, \quad RX = Y.$$

The equations (2.3) aresolved by thesuccessive substritution by columns and their solutions give us the columns of the inverse.

During the Gauss elimination the singularity is examined and the change of rows is done, if it is necessary. Menatime the determinant of the matrix is computed.

Problem 2 is to solve the system of linear equation $Ax = b$ by means of a disc by the help of Gauss elimination.

The main diagonal of the matrix plays an important role during the Gauss elimination. The places "fill in" (see [9]) in the matrix follow the main diagonal and they can be marked out before the begining of the elimination (see [1], [2],).
This possibility is used in the program, too. By this means the program uses also some zero elements in the computation, but it becomes simpler.

Let us consider the left part of the matrix. Let j be the column index of the first nonzero element in the i-th row in the left side of the diagonal. The fill in in the i-th row /left of the diagonal/ can occur between the j-th and the diagonal columns. Similarly let us consider

the upper part of the matrix. Let the i-th row index of the first nonzero element in the j-th column be in the upper side of the diagonal. The fill in in j-th column /upper side of the diagonal/ can occur between the i-th and the diagonal rows. In this manner before having begun the elimination we could mark out in the matrix the places where the nonzero elements are in the beginning of the elimination or the nonzero elements can occur during the elimination. This reasoning is applied in Program 2 and this "marked part" of the matrix is used during the computation. We suppose that the marked part of the matrix does not hold in the main memory of computer. We cut the matrix into pieces by rows such that both pieces should be contained in the main memory of the computer. The pieces of the matrix are stored in disc, they are transferred from disc to main memory and the elimination is performed between pieces in the following way: Elimination is performed in the first piece, then elimination is performed in the second piece by means of the eliminated first piece, then elimination is performed in the third piece by means of eliminated first and second pieces etc. Each eliminated piece is rewritten to disc. Finally substitution is done by means of rewritten pieces and the solution is obtained. The [1] is employed by this method for the inversion of a matrix.

Problem 3 is to solve the system of linear equation

$Ax = b$ in a band case.

Let us in the matrix

$$a_{ij} = 0, \quad \text{if} \quad |i-j| > k$$

The solution of the equation is computed by means of a disc with Gauss elimination.

The first part of the band matrix

$$A_1 = \{a_{ij}\}, \quad i,j = 1, \ldots, k$$

/that is the $(k+1) \times (k+1)$ part of the band matrix/ is stored in the main memory, the other parts are stored in disc in the following way: there are the elements

a_{ij}, $(i=k+1,\ j=2,3,\ldots,k+1)$, $(j=k+1,\ i=k, k-1, \ldots, 2)$

in the first record, there are the elements

a_{ij}, $(i=k+2,\ j=3,4,\ldots,k+2)$, $(j=k+2,\ i=k+1, k, \ldots, 3)$

in the second record, etc.

The first elimination step is the elimination of the first column of the matrix A_1 by means of the first row of the matrix A_1. Then the first row of the matrix is transferred to disc, the first column of the matrix was eliminated and the first record from disc is connected to matrix A_1. The new matrix /with size $(k+1) \times (k+1)$/ is the A_1 matrix and the previous steps are repeated in it. The difference is that after each step a new record from disc is connected to the eliminated part of the matrix. These steps are followed till each record from disc is connected to the part matrix. Then the substitution is

performed /that is the solution is obtained/ by means of the eliminated parts in disc.

Problem 4 is to solve the system of linear equations $Ax = b$ in symmetric band case. Consider in matrix

$$a_{ij} = a_{ji} \quad \text{and} \quad a_{ij} = 0 \quad , \text{ if } |i-j| > k.$$

The solution of the equation is computed by means of disc with the Choleshy method. The matrix A is factorized $A = LR$ /where L and R are triangular matrices/ by means of the Choleshy method. Then the equations

$$(2.4) \qquad Ly = b. \qquad Rx = y$$

are solved by means of substitution. The program reserves an A_1 place of size $k(k+1)/2$ in the main memory for a part of the band matrix. All the band matrices are stored in a disc. The factorization is performed in A_1 in n steps. Before each elimination step a row of band matrix from disc is connected to A_1 and after each elimination step a factorized row from A_1 is rewritten on disc. The factorised matrices are built up by means of this rewritten part. The substitution is performed by means of this rewritten part.

Problem 5 is the computation of the eigenvalues of a sparse symmetric matrix.

First the symmetrical variant of the Lanczos method is applied for a sparse symmetrix matrix A. This takes place as follows: (see [7], [10]): Starting from the normal vactor v_1 we compute

$$
\begin{aligned}
\alpha_k &= v_k^T A v_k \\
w_{k+1} &= A v_k - \alpha_k v_k - \beta_{k-1} v_{k-1}, \qquad (\beta_0 = 0) \\
\beta_k &= \| w_{k+1} \|_2 \\
v_{k+1} &= w_{k+1} / \beta_k
\end{aligned}
\tag{2.5}
$$

for $k = 1, 2, \ldots, n$. The matrix

$$
T = \begin{Vmatrix}
\alpha_1 & \beta_1 & & & \\
\beta_1 & \alpha_2 & \beta_2 & & \\
 & \ddots & \ddots & \ddots & \\
 & & & \ddots & \alpha_n
\end{Vmatrix}
\tag{2.6}
$$

is similar to matrix A. Then the eigenvalues of the matrix T are computed by means of the implicit QR method.

The application of the Lanczos - QR method for the computation of the eigenvalues of sparse matrices is very practical, because the computation of (2.5) is very easy in a sparse case too /whether in case of packed store form, or disc store/. The matrix T built up by (2.6) is tridiagonal and symmetric, therefore the application of QR method is possible in large size too. Nevertheless the method is stable.

The vectors v_k are orthonormalized vectors. However the orthogonalizm is gone wrong during the computation. Therefore a re-orthogonalization is necessary /see [10]). A new orthogonal vector is generated, if v_k becomes 0.

Problem 6 is the computation of eigenvalues os a sparse nonsymmetric matrix.

First the Lanczos method is applied for the sparse matrix A. This is the following: Let $c_0 = 0$ and vectors $x_0 = y_0 = 0$. x_1 and y_1 are optional/but $y_1^T x_1 \neq 0$/. We compute

$$
\begin{aligned}
b_k &= y_k^T A x_k / y_k^T x_k \\
x_{k+1} &= A x_k - b_k x_k - c_{k-1} x_{k-1} \\
y_{k+1} &= A^T y_k - b_k y_k - c_{k-1} y_{k-1} \\
c_k &= y_{k+1}^T x_{k+1} / y_k^T x_k
\end{aligned}
\tag{2.7}
$$

for $k = 1, 2, \ldots, n$. The matrix

$$
T = \begin{Vmatrix}
b_1 & c_1 & & & \\
1 & b_2 & c_2 & & \\
 & \ddots & \ddots & \ddots & \\
 & & & & b_n
\end{Vmatrix}
\tag{2.8}
$$

is similar to the matrix A. Then eigenvalues of the matrix T are computed by means of the bisection method /see [4], [5]). This is the following: The number of

the change of the sign in the sequence

(2.9) $1, \det(B_1 - pI_1), \det(B_2 - pI_2), \ldots, \det(B_n - pI_n)$

is equal to the number of eigenvalues which is less than p. (B_i and I_i denote the left upper minor matrices of B and I.) The computation of sequence (2.9) is very fast for matrix (2.8).

The vectors x_k and y_k are biorthogonal vectors. However the biorthogonalizm is gone wrong during the computation. Therefore a re-biorthogonalization is necessary /see [10]). A new biorthogonal vector is generated if x_k or y_k becomes 0.

3. The programs

Six programs are discussed in this chapter for the solution of the six problems which were described in the previous chapter. The constructions of the input and output dates in the programs are different, therefore we discuss those separately.

The programs have been made in FORTRAN in computer CDC 3300. We give the store need in CDC CORE nomination:
1 qp /quarter page/ = 512 words with 24 bites,
1 seg /segment/ in disc = 2o qp.

Program 1 the solution of the system of sparse linear equation $AX = B$, or inversion of a sparse matrix /see Problem 1/.

The main program reads the parameters N,IP,NY,EPS in form /3I5, E16.7/ from cards, reads the nonzero elements a_{ij} of the matrix and their indexes from file 10 in form /3/2I5,F16.7//, then prints the input dates. The negative row index after the nonzero element denotes the end of dates. During the input the program built up the array AO(·) from diagonal elements, the array A1(·) from elements outside of the diagonal and the array M1(·) from the row or column indexes of nonzero elements. The arrays A1 and M1 are in COMMON. The program uses the linked list manner for store (see [9]). After the input, the program

calls the corresponding SUBROUTINES and the solutions depending on the parameters are written out.

Parameter list:

N : order of the matrix A;

IP=0 : A=LR factorization;

IP= -1: A=LR factorization, computation and output of the inverse A^{-1} by columns;

IP=m : A=LR factorization, then m times: the right hand side is read from file 2o, in form /5E15.5/, the solution is computed and written out;

IP $<$ -1: the matrix is written out only;

NY=0 : the matrix is written out before and after the factorization;

NY=1 : the matrix is written out before the factorization;

NY=2 : the matrix is written out after the factorization;

NY=3 : the matrix is not written;

EPS : parameter for singularity investigation. If the pivot in the K-th step of elimination is less than EPS in absolute value the exchange of rows is performing. The program seeks the greatest element by column.

The SUBROUTINES:

LR factorizes the matrix A=LR;

CS performs the exchange of rows. A mark of exchange is given;

NYTAB executes the output of the matrix;

VISZ executes the back substitution to solve the equation

$$AX=B;$$

NYX prints the solution X.

The program using 84 qp in core, 5 seg in disc in CDC 3300 enables the computation of problems for $N \leq 1000$, the number of nonzeros $\leq$ 5500 /nonzeros at the start, or becoming nonzeros during computation/, the number of nonzeros in the rows $\leq$ 200.

The list of the program:

```
C
C   THE PROGRAM SOLVES THE SYSTEM OF  LINEAR EQUATIONS A*X=B WITH SPARSE MATRIX.
C      OR  INVERTS  A  SPARSE  MATRIX  BY  COLUMNS . THE  PROGRAM  COMPUTS  THE
C      DETERMINANT  OF  THE  MATRIX  IN  BOTH  CASES .
C   METHOD - CROUT#S  METHOD . THE  PROGRAM  USES  ONLY  THE  NONZERO  ELEMENTS
C      OF THE MATRIX . THE DIAGONAL ELEMENTS ARE IN  ARRAY  A0 . THE  NON  DIAGO-
C      NAL  ELEMENTS  ARE  IN  ARRAY  A1 , THEIR  ROW  OR  COLUMN  INDEXES  ARE
C      IN  ARRAY  M1 .
C
C   PARAMETERS :
C      N - NUMBER  OF  EQUATIONS ;
C      IP = 0 - THE  MATRIX  IS  FACTORIZED  TO  THE  PRODUCT  FORM  A=L*R ;
C      IP =-1 - THE  MATRIX  IS  FACTORIZED  TO  THE  PRODUCT  FORM , THE INVERSE
C               OF  THE  MATRIX  A  IS  COMPUTED  AND WRITTEN OUT  BY  COLUMNS ,
C      IP = M - THE  MATRIX  IS  FACTORIZED  TO  THE  PRODUCT  FORM  A=L*R .
C               M TIMES : THE  RIGHT-HAND-SIDE  IS  READ . THE  EQUATION  IS
C               SOLVED , THE  SOLUTIONS  ARE  WRITTEN  OUT ;
C      IP . LT. -1-  ONLY  THE  MATRIX  IS  WRITTEN  OUT .
C      THE  PARAMETER  NY  CONTROLS  THE  OUTPUT. THE MATRIX IS WRITTEN OUT IF:
C      NY = 0  BEFORE  AND  AFTER  THE  ELIMINATION ;
C      NY = 1  BEFORE  THE  ELIMINATION  ;
C      NY = 2  AFTER  THE  ELIMINATION ;
C      NY = 3  NO  WRITING  OUT .
C   EPS  CONTROLS  THE  SINGULARITY
C
      DIMENSION I1(5),J1(5),
     1A0(1000),M21(1000),M11(1000),B(1000),X(1000),MS(1000)
      COMMON /20/ A1(5500),M1(11000)
      REWIND 1
      READ 1,N,IP,NY,EPS
    1 FORMAT(3I5,F15.7)
      PRINT 1,N,IP,NY,EPS
      DO 9 I=1,N
      MS(I)=I
      M11(I)=I
      IN=I+N
    9 M21(I)=IN
      M3=2*N
    8 READ(1,10) ((I1(I),J1(I),B(I)),I=1,3)
      PRINT  2, ((I1(I),J1(I),B(I)),I=1,3)
    2 FORMAT(1H ,3(2I5,E16.7))
   10 FORMAT(3(2I5,E16.7))
      DO 4 K=1,3
      IF(I1(K)) 99,4,3
    3 I=I1(K)
      J=J1(K)
      IF(I-J) 5,5,7
    6 A0(I)=B(K)
      GOTO 4
    5 M=M11(I)
      A1(M)=B(K)
      M11(I)=M3
```

```
      M1(2*M)=J
      M1(2*M-1)=M3
      M3=M3+1
      GOTO 4
    7 M=M21(J)
      A1(M)=B(K)
      M21(J)=M3
      M1(2*M)=I
      M1(2*M-1)=M3
      M3=M3+1
    4 CONTINUE
      GOTO 8
   99 CONTINUE
      IF (IP.GT.-2) GOTO 11
      CALL NYTAB(N,M3,A0)
      GOTO 98
   11 CONTINUE
      IF(NY.GT.1) GOTO 90
      CALL NYTAB(N,M3,A0)
   90 CONTINUE
      CALL LR(N,M3,A0,M11,M21,EPS,MS,3,X)
      IF(NY.EQ.1.OR.NY.EQ.3) GOTO 91
      CALL NYTAB(N,M3,A0)
   91 CONTINUE
      IF(IP.EQ.0) GOTO 98
      IF(IP.GT.0) GOTO 20
      DO 39 M=1,N
      DO 31 I=1,N
   31 B(I)=0.
      MM=MS(M)
      B(MM)=1.
      CALL VISZ(N,MM,IP,A0,3,X)
      CALL NYX(N,M,X)
   39 CONTINUE
   20 REWIND 2
      DO 29 M=1,IP
C
C    THE RIGHT HAND SIDE IS READING
C
      READ(2,21) (B(I),I=1,N)
   21 FORMAT(5E15.5)
      CALL VISZ(N,M,IP,A0,3,X)
      CALL NYX(N,M,X)
   29 CONTINUE
   98 CONTINUE
      END
```

```
      SUBROUTINE LR(N,M3,A0,M11,M21,EPS,MS,A5,A6)
C
C  THE SUBROUTINE LR FACTORIZED THE MATRIX TO THE PRODUCT FORM A=L*R
C
      DIMENSION A0(N),M11(N),M21(N),MS(N),A5(N),A6(N),
     1M6(200),M5(200)
      COMMON /20/ A1(5500),M1(11000)
      N1=N-1
      DET=1.
      DO 19 K=1,N1
C
C  IF THE PIVOT ELEMENT IN MODUL .LE. EPS THEN NEW PIVOT ELEMENT IS
C     CHOSEN BY COLUMN AND EXCHANGE OF ROWS IS DONE .
C
      IF (ABS(A0(K)).GT.EPS) GO TO 19
      CALL CS(K,N,M3,A0,M11,M21,MS,EPS)
      DET=-DET
   19 K1=M1(2*K-1)
      K5=K
      K2=1
    1 IF(K1.EQ.0) GOTO 2
      IF(M1(2*K5).LE.K) GOTO 22
      M5(K2)=M1(2*K5)
      A1(K5)=A1(K5)/A0(K)
      A5(K2)=A1(K5)
      K2=K2+1
   22 K5=K1
      K1=M1(2*K1-1)
      GOTO 1
    2 KA=K+N
      K1=M1(2*KA-1)
      K5=KA
      K3=K2-1
      K2=1
    6 IF(K1.EQ.0) GOTO 5
      M6(K2)=M1(2*K5)
      A6(K2)=A1(K5)
      K5=K1
      K1=M1(2*K1-1)
      K2=K2+1
      GOTO 6
    5 K4=K2-1
      DO 7 I=1,K3
      DO 7 J=1,K4
      MI=M5(J)
      MJ=M5(I)
      IF(MI.NE.MJ) GOTO 9
      A0(MI)=A0(MI)-A5(I)*A6(J)
      GOTO 7
    9 IF(MI.GT.MJ) GOTO 11
      K5=MI
      K1=M1(2*MI)
```

```
      K2=M1(2*MI-1)
   13 IF(K2.EQ.0) GOTO 14
      IF(K1.EQ.MJ) GOTO 12
      K5=K2
      K1=M1(2*K2)
      K2=M1(2*K2-1)
      GOTO 13
   12 A1(K5)=A1(K5)-A5(I)*A6(J)
      GOTO 7
   14 L=M11(MI)
      A1(L)=-A5(I)*A6(J)
      M1(2*L)=MJ
      M1(2*L-1)=MB
      M11(MI)=M3
      M3=M3+1
      GOTO 7
   11 MA=MI+N
      K1=M1(2*MA)
      K2=M1(2*MA-1)
      K5=MA
   15 IF(K2.EQ.0) GOTO 16
      IF(K1.EQ.MI) GOTO 17
      K5=K2
      K1=M1(2*K2)
      K2=M1(2*K2-1)
      GOTO 15
   17 A1(K5)=A1(K5)-A5(I)*A6(J)
      GOTO 7
   16 L=M21(MJ)
      A1(L)=-A5(I)*A6(J)
      M1(2*L)=MI
      M1(2*L-1)=MB
      M21(MJ)=M3
      M3=M3+1
    7 CONTINUE
   10 DET=DET*A0(K)
      DET=DET*A0(N)
      PRINT 18,DET
   18 FORMAT(////7H   DET=,E15.7)
      RETURN
      END
```

```
      SUBROUTINE CS(K,N,M3,A0,M11,M21,MS,EPS)
C
C  THE  SUBROUTINE  CS  PERFORMS  THE  EXCHANGE OF  ROWS.
C
      DIMENSION A0(N),M11(N),M21(N),MS(N)
      COMMON /20/ A1(5500),M1(11000)
      KK=K
      A=ABS(A0(K))
      J=K+N
    2 J1=M1(2*J-1)
      IF(J1.EQ.0) GOTO 3
      IF(A.GE.ABS(A1(J))) GOTO 1
      A=ABS(A1(J))
      KK=J
    1 J=J1
      GOTO 2
    3 IF(A.LT.EPS) GOTO 21
C
C  THE  PIVOT  ELEMENT  IS  CHOSEN  FROM  J2-TH  ROW .
C
      J2=M1(2*KK)
      AK=A0(K)
      A0(K)=A1(KK)
      A1(KK)=AK
      M1(2*KK)=J2
      MS1=MS(K)
      MS(K)=MS(J2)
      MS(J2)=MS1
      AJ=A0(J2)
      A0(J2)=0
      K1=K-1
      DO 4 I=1,K1
      I4=I+N
    6 I1=M1(2*I4-1)
      I3=0
      IF(I1.EQ.0) GOTO 4
      IF(M1(2*I4).EQ.K) I3=I4
      IF(M1(2*I4).EQ.J2) M1(2*I4)=K
      IF(I3.NE.0) M1(2*I4)=J2
      I4=I1
      GOTO 6
    4 CONTINUE
      I=K
   11 I1=M1(2*I-1)
      IF(I1.EQ.0) GOTO 10
      I2=M1(2*I)
      IF(I2-J2) 7,8,13
    8 A0(J2)=A1(I)
   13 I=I1
      GOTO 11
    7 I4=I2+N
   15 I5=M1(2*I4-1)
```

```
      IF(I5.EQ.0) GOTO 19
      IF(M1(2*I4).NE.J2) GOTO 14
      A=A1(I4)
      A1(I4)=A1(I)
      I5=M11(J2)
      M11(J2)=M3
      M1(2*I5-1)=M3
      M3=M3+1
      M1(2*I5)=I2
      A1(I5)=A
      GOTO 13
   19 I5=M21(I2)
      M21(I2)=M3
      M1(2*I5-1)=M3
      M3=M3+1
      M1(2*I5)=J2
      A1(I5)=A1(I)
      GOTO 13
   14 I4=M1(2*I4-1)
      GOTO 15
   10 I5=M11(J2)
      M11(J2)=M3
      M1(2*I5-1)=M3
      M1(2*I5)=J2
      M3=M3+1
      A1(I5)=AJ
      K1=K+1
      K2=J2-1
      DO 23 K3=K1,K2
      I=K3+N
      J1=K
   25 IF(M1(2*J-1).EQ.0) GOTO 23
      IF(M1(2*J).NE.J2) GOTO 24
   27 IF(M1(2*J1).EQ.K3) GOTO 23
      IF(M1(2*J1).GE.I2) GOTO 31
      IF(M1(2*J1-1).EQ.0) GOTO 26
   31 J1=M1(2*J1-1)
      GOTO 27
   24 J=M1(2*J-1)
      GOTO 25
   26 I5=M11(J2)
      M11(J2)=M3
      M1(2*I5-1)=M3
      M3=M3+1
      M1(2*I5)=K3
      A1(I5)=A1(J)
      A1(J)=0
   23 CONTINUE
      L1=M1(2*K-1)
      L2=M1(2*K)
      M1(2*K-1)=M1(2*I2-1)
      M1(2*K)=M1(2*I2)
```

```
      M1(2*I2-1)=L1
      M1(2*I2)=L2
      A=A1(K)
      A1(K)=A1(J2)
      A1(J2)=A
      I=J2
   28 I1=M1(2*I)
      IF(I1.EQ.0) GOTO 29
      IF(I1.GT.J2) GOTO 32
      A1(I)=0
   32 I=M1(2*I-1)
      GOTO 28
   29 CONTINUE
      GOTO 16
   21 CONTINUE
      PRINT 20,N,K
   20 FORMAT (12H  SINGULARIS.2I5//)
      STOP
   16 CONTINUE
C
C  THE  MARK  OF  EXCHANGE .
C
      PRINT 33,K,J2
   33 FORMAT(7H OSERE:,2I5)
      I=M11(J2)
      M11(I2)=M11(K)
      M11(K)=I
      RETURN
      END
```

```
      SUBROUTINE NYTAB(N,M3,A0)
C
C  THE  SUBROUTINE  NYTAB  EXECUTES  THE  OUTPUT  OF  MATRIX
C
      DIMENSION A0(N)
      COMMON /20/ A1(5500),M1(11000)
      M5=2*M3
      PRINT 1,N
      PRINT 2,(A0(I),I=1,N)
      PRINT 2,(A1(I),I=1,M3)
      PRINT 3,(I,I=1,10)
      PRINT 1,(M1(I),I=1,M5)
    3 FORMAT(/10I9/)
    1 FORMAT(10(I5,I4))
    2 FORMAT(1H ,10F10.5)
      RETURN
      END
```

```
      SUBROUTINE VIS7(N,M,IP,A0,B,X)
C
C  THE  SUBROUTINE  VIS7  EXECUTES  THE  BACK-SUBSTITUTION TO SOLVE  THE
C    EQUATION  A*X=B.
C
      DIMENSION A0(N),B(N),X(N)
      COMMON /20/ A1(5500),M1(11000)
      DO 1 I=1,N
    1 X(I)=0
      K=1
      IF(IP.LT.0) K=M
      DO 2 J=K,N
      X(J)=B(J)/A0(J)
      J2=J+N
    3 J1=M1(2*J2-1)
      J2=M1(2*J2)
      IF(J1.EQ.0) GOTO 2
      B(J2)=B(J2)-A1(J3)*X(J)
      J2=J1
      GOTO 3
    2 CONTINUE
      N1=N-1
      DO 5 I=1,N
    5 B(I)=X(I)
      DO 4 I=1,N1
      I1=N-I
      J=I1
    6 J1=M1(2*I-1)
      J2=M1(2*J)
      IF(J1.EQ.0) GOTO 4
      B(I1)=B(I1)-A1(J1)*X(J2)
      J=J1
      GOTO 6
    4 X(I1)=B(I1)
      RETURN
      END
```

```
      SUBROUTINE NYX(N,M,X)
C
C  THE  SUBROUTINE  NYX  PRINTS  THE  SOLUTION
C
      DIMENSION X(N)
      PRINT 1,M
      PRINT 2,(X(I),I=1,N)
    1 FORMAT(///I5)
    2 FORMAT(5E13.9)
      RETURN
      END
```

Program 2 is for solution of a system of sparse linear equation by means of disc /see Problem 2/.

The program reads from file 10 the parameters N,N9,EPS in form /2I9, E15.5/, then the nonzero elements of the matrix and their indexes from file 2o in form /3/E16.9, 2I5/. The nonzero elements should be stored in file 2o by rows. After the last element there must be one negative row index. Finally the right-hand side is read from file 3o in form /5E15.9/.

The program during the reading by means of SUBROUTINE RENDEZ built up the sparse matrix in disc. During the computation the pieces of the matrix as unit are transferred between the main memory and disc memory. The program calls the SUBROUTINE SLED which computes the solution. Then the solution is written out.

Parameter list:

N : the number of the equations;

N9 : the maximal size of one piece;

EPS : parameter to investigate the singularity.

The program using 24 qp in core and 55 seg in disc is suitable for the solution of an equation with $N \leq 1000$, $N9 < 4000$, number of pieces of nonzero elements ≤ 100.

The list of the program:

```
C  THE PROGRAM SOLVES THE SPARSE LINEAR SYSTEM WITH HELP OF THE
C     SUBROUTINE SLED. THE PROGRAM CARRIES OUT THE INPUT FROM
C     FILE 20,BY MEANS OF SUBROUTINE RENDEZ
C     CONSTRUCTS THE MATRIX IN DISC , CALLS THE SOUBROUTINE SLED AND
C     PRINTS THE SOLUTION.
C  PARAMETERS :
C     N - THE NUMBER OF EQUATIONS ;
C     N9 - MAX LENGTH OF A PART OF THE MATRIX ;
C     EPS - THE PARAMETER FOR INVESTIGATION OF THE SINGULARITY
C  THE PROGRAM USES THE FOLLOWING ARRAYS:
C     M1(I) - THE NUMBER OF NONZERO ELEMENTS LEFT TO THE DIAGONAL IN THE
C             I- TH ROW ;
C     M2(I) - THE NUMBER OF NONZERO ELEMENTS RIGHT TO THE DIAGONAL IN
C             THE I -TH ROW ;
C     ID(K)+1 IS THE FIRST ROW INDEX IN THE K-TH PART OF THE MATRIX;
C     ID(K+1) IS THE LAST ROW NUMBER IN THE K -TH PART OF THE MATRIX;
C     M(J) : THE COLUMN INDEX OF THE NONZERO ELEMENTS RIGHT TO THE
C            DIAGONAL, SUCCESSIVELY BY ROWS .
C
C  THE PROGRAM USES SUCH A APRT OF MATRIX IN WHICH THE ELEMENTS ARE NON-
C     ZERO, OR NONZERO ELEMENTS CAN APPEAR DURING THE ELIMINATION .
C
      DIMENSION M1(1000),M2(1000),M3(1000),M4(1000),M(3000),ID(101),I3(
     13),J1(3),IM(101) ,IA(101)
     2,A(4000),B(1000),X(1000)
      COMMON /10/ A,B,X,M
      REWIND 10
      READ(10,11) N,N9,EPS
      PRINT 11,N,N9,EPS
      REWIND 20
      CALL RENDEZ (N,N9,M1,M2,M3,M4,ID,I3,J1,IM,IA)
      REWIND 30
      IM1=1
      M3(1)=1
      M4(1)=1
      DO 41 I=2,N
      M3(I)=M3(I-1)+M2(I-1)+M1(I)+1
   41 CONTINUE
      M41=2
      DO 112 I=2,N
      IF (I.LE.ID(M41)) GOTO 114
      M4(I)=1
      M41=M41+1
      GOTO 112
  114 M4(I)=M4(I-1)+M2(I-1)
  112 CONTINUE
      DO 34 K=1,N9
   34 A(K)=0
C
C    INPUT OF THE RIGHT SIDE . PERFORMATION BY FORMAT 13.
C
      READ(30,13) (B(I),I=1,N)
```

```
      ENDFILE 1
      REWIND 1
      ENDFILE 3
      REWIND 3
      ENDFILE 4
      REWIND 4
      CALL SLED(N,ID,M1,M2,M3,M4,NO,IM,IA,EPS)
C
C    OUTPUT  OF  THE  SOLUTION
C
      PRINT 27
      PRINT 30,(I,X(I),I=1,N)
   27 FORMAT(14H1THE SOLUTION://)
   11 FORMAT(2I8,E15.5)
   13 FORMAT(5(E15.9))
   30 FORMAT(I5,E18.9)
      END
```

```
      SUBROUTINE RENDEZ (N,N9,M1,M2,M3,M4,ID,I3,J1,IM,IA)
      DIMENSION M1(1),M2(1),M3(1),M4(1),ID(1),I3(1),J1(1),IM(1),IA(1)
       COMMON /10/ A(4000),B(1000),X(1000),M(3000)
C
C  THE SUBROUTINE RENDEZ BUILTS UP THE MATRIX IN DISC AND THE ARRAYS
C  M1,M2,ID,IA AND IM  BY MEANS OF INPUT.
C
      L=1
      ID1=1
      ID(ID1)=L-1
      K=0
      KA=0
      I8=0
      K3=1
C
C   INPUT  OF  NONZERO  ELEMENTS : A(I,J), I, J,  WHERE  I  IS  THE  ROW , J  IS
C     THE  COLUMN  INDEX .  PERFORMATION  BY  FORMAT  9, THE  END  MARK  IS I<0.
C
    6 READ (20,9) (X(I),I3(I),J1(I),I=1,3)
      PRINT 9,(X(I),I3(I),J1(I),I=1,3)
      DO  1 I=1,3
      IF (I3(I)) 25,1,2
    2 IF (I3(I)-L) 19,3,10
    3 J2=J1(I)
      J3=I3(I)-J2
      IF(J3) 5,4,4
    4 B(J2)=X(I)
      IF (M1(L).LT.J3) M1(L)=J3
      GOTO 1
    5 K1=L+K3
      I8=I8+1
      K3=K3+1
      B(K1)=X(I)
      M3(K1)=J2
    1 CONTINUE
      GOTO 6
   10 M2(L)=I8
      K5=K+1
      L1=L+1
      L2=M1(L)+1
      DO 7 J=1,L2
      KA1=KA+J
      KB=L-M1(L)+J-1
    7 A(KA1)=B(KB)
      KA=KA+L2
      I8=0
      GOTO 26
   19 PRINT  18,L,I3(I),J1(I)
      GOTO 1
   25 I9=-1
      GOTO 10
   26 I4=L+1
```

```
      J4=L
      IF(M4(L).EQ.L) J4=L+1
   14 IF(M3(I4).EQ.0.AND.M4(J4).EQ.0) GOTO 8
      K=K+1
      IF (M3(I4) - M4(J4)) 11,12,13
   12 M(K)=M3(I4)
      KA = KA+1
      A(KA)=B(I4)
      I4=I4+1
      J4=J4+1
      GOTO 14
   13 IF(M4(J4).NE.0) GOTO 15
   16 KA=KA+1
      A(KA)=B(I4)
      M(K)=M3(I4)
      I4=I4+1
      GOTO 14
   15 M(K)=M4(J4)
      KA=KA+1
      J4=J4+1
      GOTO 14
   11 IF(M3(I4).EQ.0) GOTO 15
      GO TO 16
    8 DO 17 J=L,N
      M3(J)=0
   17 M4(J)=0
      L=L+1
      DO 20 J=K5,K
      L2=L1+J-K5
   20 M4(L2)=M(J)
      K3=1
      DO 33 J=1,N
   33 B(J)=0.
      IF(KA.LT.N9.AND.I9.EQ.0) GOTO 3
      IA(ID1)=KA
      IM(ID1)=K
      WRITE (1)(A(J),J=1,KA)
      WRITE (3)(M(J),J=1,K)
      WRITE (4)(M(J),J=1,K)
      DO 24 J=1,KA
   24 A(J)=0
      ID1=ID1+1
      ID(ID1)=L-1
      DO 28 J=1,K
   28 M(J)=0
      K=0
      KA=0
      K3=1
      IF(L.LT.N.AND.I9.EQ.0) GOTO 3
    9 FORMAT (3(F16.9,2I5))
   18 FORMAT (12H ORDER ERROR,3I5)
      RETURN
      END
```

```
      SUBROUTINE  SLED(N,ID,M1,M2,M3,M4,N9,IM,IA,EPS)
C
C   THE  SUBROUTINE  SLED  SOLVES  THE  SPARSE  LINEAR  SYSTEM .  IT  PERFORMS
C      THE  GAUSS  ELIMINATION  AMONG  PARTS OF THE SPARSE MATRIX .  THE  WHOLE
C      MATRIX  IS  IN  DISC .  PARTS  ARE  TRANSFERRED BETWEEN DISC  AND  MAIN
C      MEMORY .
C
      DIMENSION M1(1),M2(1),M3(1),M4(1),ID(1)
     1,A(4000),B(1000),X(1000),D(4000)
     2,MA(3000),MD(3000),IM(1),IA(1)
      COMMON /10/ A,B,X,MA
      J1=1
      IM1=1
      IM2=1
      J2=0
    1 IA1=IA(IM1)
      READ(1) (A(I),I=1,IA1)
      IM3=IM(IM1)
      IM1=IM1+1
      READ (3) (MA(I),I=1,IM3 )
      I3=ID(J1)+1
      I4=ID(J1+1)
      IF(J1.EQ.1) GO TO 78
      J2=M3(I3-1)+M2(I3-1)
   78 J5=J1-1
      IF (J5) 3,3,4
    4 DO 29 J4=1,J5
      I5=ID(J4)+1
      I6=ID(J4+1)
      IA1=IA(IM2)
      READ(2) (D(I),I=1,IA1)
      IM3=IM(IM2)
      IM2=IM2+1
      READ (4) (MD(I),I=1,IM3)
      DO 29 J6=I5,I6
      J7=0
      IF(J4.EQ.1) GO TO 79
      J7=M3(I5-1)+M2(I5-1)
   79 DO 29 L2=I3,I4
      L3=J6+M1(L2)-L2
      L4=I5+L3
      IF(L3) 29,50,50
   50 L5=M3(L2)+J6-L2-J2
      L7=M2(J6)
      DO 81 J=1,L7
      K=M4(J6)+J-1
      L6=MD(K)
      L14=M3(J6)+J-J7
      IF (L2-L6) 30,31,32
   30 L8=M2(L2)
      DO 33 L=1,L8
      L9=M4(L2)+L-1
```

```
      IF(MA(L9)-L6) 33,34,34
   33 CONTINUE
   34 K1=M3(L2)+1
      GOTO 35
   31 K1=M3(L2)
      GOTO 35
   32 K1=M3(L2)+L6-I2
   35 K1=K1-I2
      A(K1)=A(K1)-A(L5)*B(L14)
   81 CONTINUE
      B(L2)=B(L2) -A(L5)*B(L6)
   29 CONTINUE
    3 DO 49 L2=I3,I4
      K1=M3(L2)-J2
      IF(ABS(A(K1)).LT.EPS) GOTO 85
      B(L2)=B(L2)/A(K1)
      I13=L2+1
      DO 110 I=I13,I4
      IF(I-L2.GT.M1(I)) GOTO 110
      L6=M3(I)+L2-I-J2
      B(I)=B(I)-A(L6)*B(L2)
  110 CONTINUE
      L4=M2(L2)
      DO 49 L=1,L4
      L5=M3(L2)+L-J2
      A(L5)=A(L5)/A(K1)
      DO 49 I=I13,I4
      IF(I-L2.GT.M1(I)) GOTO 49
      L6=M3(I)+L2-I-J2
      L7=M4(L2)+L-1
      L8=MA(L7)
      IF(I-L8) 38,39,40
   39 L7=M3(I)
      GOTO 41
   40 L7=M3(I)+L8-I
      GOTO 41
   38 L9=M2(I)
      DO 42 J=1,L9
      L1=M4(I)+J-1
      IF(MD(L1)-L8) 42,43,43
   42 CONTINUE
   43 L7=M3(I)+J
   41 L7=L7-J2
      A(L7)=A(L7)-A(L6)*A(L5)
   49 CONTINUE
      IF(ID(J1+1)-N) 46,47,47
   46 IA1=IA(IM2)
      WRITE(2) (A(I),I=1,IA1)
      J1=J1+1
      ENDFILE 2
      REWIND 2
      REWIND 4
```

```
      IM2=1
      GOTO 1
   47 ENDFILE 2
      IM3=IM(IM2)
      READ(4) (MD(I),I=1,IM3)
      GOTO 99
   69 CALL BACKSTEP(1H2)
      CALL BACKSTEP(1H2)
      CALL BACKSTEP(1H4)
      CALL BACKSTEP(1H4)
      IA1=IA(IM2)
      IM3=IM(IM2)
      READ(2) (A(I),I=1,IA1)
      READ (4) (MD(I),I=1,IM3)
      IM2=IM2-1
   99 L3=ID(J1)+1
      L4=ID(J1+1)
      L5=0
      IF(L3.LE.1) GO TO 76
      L5=M3(L3-1)+M2(L3-1)
   76 L1=M3(N)-L5
      IF (L9) 63,64,64
   64 L9=-1
      X(N)=B(N)
      L4=L4-1
   63 DO 68 I=L3,L4
      K1=L4-I+L3
      X(K1)=B(K1)
      S=0
      L8=M2(K1)
      L7=M4(K1)
      DO 68 J=1,L8
      L6=L7+J-1
      L11=M3(K1)+J-L5
      I1=MD(L6)
   68 X(K1)=X(K1)-A(L11)*X(I1)
      J1=J1-1
      IF(J1) 67,67,69
   85 PRINT 86,L2,I3,I4,K1,A(K1)
   86 FORMAT(10H SINGULAR:,4I5,E15.5)
   67 RETURN
      END
```

Program 2 is for the solution of a system of linear equation with band matrix /see Problem 3/.

The solution is computed in SUBROUTINE SZALAG. We can call it:

CALL SZALAG (N,K,DPS),

where N is the number of equation. The bandwidth is $2K+1$. The parameter DPS investigates the singularity. If the pivot element in absolute value is less than DPS a mark is written out and the computation stops. There is a statement

COMMON /1/ A, B

in subroutine, where $A(\cdot)$ is an array for the left upper part of the band matrix /the size is $(K+1)\times(K+1)$/, $B(\cdot)$ is an array for the right hand size of equation. The other parts of the band matrix should be written in the file 1 in disc such as it was discussed in Problem 3. The solution is computed in array B and it is written out.

The subroutine using 88 qp in core and 200 seg in disc is suitable for solving an equation with N≤7000, K≤100, /but $N.K < 3.3\ 10^5$/.

The list of the program:

```
      SUBROUTINE SZALAG(N,K,EPS)
C
C  THE SUBROUTINE SZALAG SOLVES A SYSTEM OF EQUATION WITH BAND MATRIX .
C     THE FIRST (K+1)*(K+1) PART OF BAND IS IN ARRAY A AT THE BEGINING
C     THEN ONE ROW AND COLUMN ARE TRANSFERRED FROM FILE 1 SUCCESSIVELY
C     AND IT WILL BE CONNECTED WITH MATRIX A .
C     AFTER EACH ELIMINATION STEP ONE ROW IS WRITTEN TO THE FILE 2.
C     THE BACK-SUBSTITUTION IS CARRIED OUT WITH HELP OF FILE 2. THE RIGHT HAND
C     SIDE IS IN VECTOR B. ARRAYS A AND B ARE IN COMMON .
C  PARAMETER EPS LOOKS FOR SINGULARITY . IN CASE OF SINGULARITY A CERTAIN
C     MARK IS WRITTEN OUT .
C  PARAMETERS : N  IS THE NUMBER OF EQUATIONS ;
C               K  IS THE HALF WIDTH OF BAND .
C     THE SOLUTION IS FORMED IN ARRAY B .
C
      DIMENSION A(101,101),B(7000),C(201)
      COMMON /1/ A,B
      N1=N-1
      K1=K+1
      K2=K+K1
      REWIND 1
      DO 10 L=1,N1
      IF(ABS(A(1,1)).LT.EPS) GOTO 11
      A1=1/A(1,1)
      DO 1 J=2,K
    1 A(1,J)=A1*A(1,J)
      B(L)=A1*B(L)
      DO 2 I=2,K1
      I1=L+I-1
      B(I1)=B(I1)-A(I,1)*B(L)
      DO 2 J=2,K
    2 A(I,J)=A(I,J)-A(I,1)*A(1,J)
      WRITE(2) (A(1,J),J=2,K1)
      DO 3 I=1,K
      DO 3 J=1,K
    3 A(I,J)=A(I+1,J+1)
      DO 4 I=1,K1
      A(K1,I)=0
    4 A(I,K1)=0
      IF(L-N+K) 5,10,10
    5 READ (1) (C(I),I=1,K2)
      DO 9 I=1,K
      A(K1,I)=C(I)
      J=K2-I+1
    9 A(I,K1)=C(J)
      A(K1,K1)=C(K1)
   10 CONTINUE
      L=N
      IF(ABS(A(1,1)).LT.EPS) GO TO 11
      B(N)=B(N)/A(1,1)
      ENDFILE 2
      CALL BACKSTEP(1H2)
```

```
      DO 6 L=1,N1
      L1=N-L
      CALL BACKSTEP(1H2)
      READ(2) (A(1,I),I=1,K)
      CALL BACKSTEP(1H2)
      DO 6 I=1,K
      I1=L1+I
    6 B(L1)=B(L1)-A(1,I)*B(I1)
      GOTO 13
   11 PRINT 12,L,A(1,1)
   12 FORMAT(14H SZING.     :,I5,E13.5)
   13 CONTINUE
      RETURN
      END
```

Program 4 is for the solution of a system of the linear equations with symmetric band matrix /see Problem 4/.

The program reads the parameters N,MK,EPS in form /2I5, F10/ from cord, where N is the half width of band, MK is the number of equations, EPS investigates the singularity. If the pivot in Cholesky factorization is less then EPS in absolute value a mark is written out and the computation stops. The program reads in the way of rows the right parts of matrix from file 1, builts up in a one-dimensional array A /a $(N+1)\times(N+1)$ size part of the band matrix/ and applies the Cholesky factorization for this part. After each step one row of the factorised matrix is rewritten into file 2 in disc. After MK steps the factor matrix is formed in file 2. Then the program reads the right hand side from file 3 and by means of (2.4) the solution is computed. The solution appears in array B and it is written out.

The program using 83 qp in core and 200 seg in disc is suitable for the solution of an equation with $MK \leq 16110$, $N \leq 179$, /but $N.MK < 10^6$/.

The list of the program:

```
      DIMENSION A(16110),B(16110),M(179),C(179)
      EQUIVALENCE (A(1),B(1))
      COMMON /10/ A
C
C  THE PROGRAM SOLVES A SYSTEM OF LINEAR EQUATION WITH
C  SYMMETRIC BAND MATRIX OF MEAN OF CHOLEVSKY METHOD
C
      READ 22,NK,N,EPS
   22 FORMAT(2I5,F10.0)
      NN=N*(N+1)/2
      REWIND 1
      REWIND 3
      N2=N+1
      N3=NK+N
      M(1)=0
      M1=N
      DO 1 K=2,N
      M(K)=M(K-1)+M1
    1 M1=M1-1
      DO 2 I=1,NN
    2 A(I)=0
      DO 10 KK=1,NK
      READ(1) (A(I),I=1,N1)
      N1=N-1
      S=A(1)
      DO 3 I=2,N
      M1=M(I)+1
    3 S=S-A(M1)*A(M1)
      S=SQRT(S)
      A(1)=S
      IF(S.LT.EPS) GOTO 9
      DO 4 I=2,N
      I1=N-I+1
      S1=A(I)
      DO 5 J=2,I1
      M1=M(J)+1
      M2=M(J)+I
    5 S1=S1-A(M1)*A(M2)
    4 A(I)=S1/S
      DO 6 I=1,N
      M1=M(I)+1
    6 C(I)=A(M1)
      WRITE(2) (C(I),I=1,N)
      DO 7 K1=1,N1
      DO 7 L=1,K1
      K=N-K1
      I=M(K)+L+1
      J=M(K+1)+L
    7 A(J)=A(I)
   10 CONTINUE
      ENDFILE 2
      DO 8 I=1,N
```

```
 8 B(I)=0
   READ(3) (B(I),I=N2,N3)
   REWIND 2
   DO 11 K=1,NK
   READ(2) (C(I),I=1,N)
   K1=N+K
   S=B(K1)
   DO 12 I=2,N
   K2=K1-I+1
12 S=S-B(K2)*C(I)
11 B(K1)=S/C(1)
   DO 13 K=1,NK
   CALL BACKSTEP(1H2)
   READ(2) (C(I),I=1,N)
   K1=N3-K+1
   CALL BACKSTEP(1H2)
   B(K1)=B(K1)/C(1)
   DO 14 I=1,N1
   K2=K1-I
14 B(K2)=B(K2)-B(K1)*C(I+1)
13 CONTINUE
   PRINT 16,N,NK
16 FORMAT(12H A MEGOLDAS:,/2I5//)
   PRINT 17,(B(I),I=N2,N3)
17 FORMAT(10F12.5)
   GOTO 18
 9 PRINT 19,KK,S
19 FORMAT(13H SZINGULARIS:,T5,E15.5)
18 CONTINUE
   END
```

Program 5 is for the computation of eigenvalues of a symmetric sparse matrix /see Problem 5/.

The program reads the parameters N, EPS and EPS2 from card in form /I4,2E16.8/, where N is the order of matrix, EPS is the accuracy of eigenvalues and EPS2 investigates the singularity. Then reads N+1 number into array IS from file 2 in form /15I5/, which points to the beginning of the rows in arrays A and M. The nonzero elements of the right-hand side of the symmetric matrix and their column indexes are read from file 1 in form (4(I4,E16.8)) into array M and A. The nonzero elements in file 1 should be arranged by rows. The end of nonzero elements is denoted by a negative index.

The program executes the symmetric Lanczos method. The symmetric tridiagonal matrix is formed in array E(.) and D(.) . The SUBROUTINE AV (N,V,S) is used for the computation of product S=AV. After finishing the Lanczos method the explicite QR method is used for computation of eigenvalues of the tridiagonal matrix. It is done in SUBROUTINE QR (N, SMACHEPS,E,D,IER) . The eigenvalues of the matrix are formed in array B(.) . The serial number, the accuracy of eigenvalues and the eigenvalues are written out. The SUBROUTINE ORTSIM perform the re-orthogonalization and new orthogonal vector v_k is generated, if $\|v_k\| \leq$ EPS2.

The program using 65 qp in core and 65 seg in disc is suitable for the computation of the eigenvalues of a matrix with $N \leq 500$ and the number of nonzero elements ≤ 5000.

The list of the program:

```
C
C   THE PROGRAM COMPUTS THE EIGENVALUES OF A SYMMETRIC SPARSE MATRIX.
C   THE SYMMETRIC LACZOS METHOD  THEN  THE  IMPLICIT QR METHOD  ARE USED.
C
      DIMENSION C(500),B(500),V(500),W(500),S(500),I1(5),A1(5)
      COMMON /10/ M(5000),A(5000),IS(500)
      REWIND 1
      REWIND 2
C
C   THE INPUT OF PARAMETERS:
C   N  -  IS THE ORDER OF MATRIX
C   EPS  -  IS THE ACCURACY FOR EIGENVALUES
C   EPS2 INVESTIGATES THE SINGULARITY
C
      READ  17,N,EPS,EPS2
      PRINT 17,N,EPS,EPS2
      N1=N+1
C
C   THE ARRAY ELEMENT IS(I) LOOKS FOR THE FIRST NONZERO ELEMENT IN ROW I.
C
      READ(2,16) (IS(I),I=1,N1)
      K=1
C
C   THE NONZERO ELEMENTS AND THEIR COLUMN INDEXES ARE READ FROM FILE 1
C
      CALL MILCO(IDO)
      PRINT 30,IDO
   30 FORMAT(3I10)
    1 READ(1,15) (I1(I),A1(I),I=1,4)
      DO 2 I=1,4
      IF(I1(I)) 10,2,4
    4 M(K)=I1(I)
      A(K)=A1(I)
      K=K+1
    2 CONTINUE
      GOTO 1
   10 DO 5 K=1,N
    5 V(K)=0.
      L3=1
      NK=1
      PRINT 16,(IS(I),I=1,N1)
      NN1=IS(N+1)-1
      PRINT 15,(M(I),A(I),I=1,NN1)
      V(1)=1
      V(N1)=1.
  102 CONTINUE
      WRITE(3) (V(I),I=1,N1)
      C(1)=0
      DO 6 K=1,N
      NK1=K+1
      L4=1
      CALL AV(N,V,S)
      DO 7 L=1,N
    7 B(K)=B(K)+V(L)*S(L)
      DO 8 L=1,N
    8 W(L)=S(L)-B(K)*V(L)-C(K)*W(L)
```

```
      CALL ORTSIM(N,K,W,S,C,EPS2,T,L3,L4)
      IF(L4.LT.0) GOTO 106
      DO 11 L=1,N
      D=V(L)
      V(L)=W(L)/T
   11 W(L)=D
    6 CONTINUE
  106 I5=NK
      NK=K+1
      C(K+1)=0
      PRINT 13,(I,C(I),B(I),I=I5,NK1)
      NK2=NK1-I5
      DO 121 I=1,NK2
      I6=I5+I-1
      C(I+1)=C(I6+1)
  121 B(I)=B(I6)
      CALL QR(NK2,EPS,C,B,IER)
C
C  THE EIGENVALUES AND THEIR ACCURACY ARE WRITTEN OUT
C
      DO 124 I=1,NK2
      I6=NK2-I+I5
      I7=NK2-I+1
      C(I6)=C(I7)
  124 B(I6)=B(I7)
      NK3=NK2-1+I5
      PRINT 123,(I,C(I),B(I),I=I5,NK3)
      PRINT 130
      IF(K.GE.N) STOP
C
C  A NEW ORTHOGONAL VECTOR IS GENERATED , IF  W(.) IS SMALL
C
      L3=L3+1
      L2=L3
      DO 120 I=1,N
      C(I)=0.
  120 B(I)=0.
      IF(L3.GT.N) GOTO 28
      DO 111 L=L2,N
      DO 12 I=1,N
   12 W(I)=0
      W(L)=1.
      REWIND 3
      DO 113 I=1,K
      READ(3) (S(J),J=1,N1)
      U=0
      DO 114 J=1,N
      U=U+S(J)*W(J)
  114 CONTINUE
      Z=U/S(N1)
      DO 105 J=1,N
  105 W(J)=W(J)-Z*S(J)
  113 CONTINUE
```

```
      F=0
      DO 116 I=1,N
  116 F=F+W(I)*W(I)
      T=SQRT(F)
      DO 117 I=1,N
  117 V(I)=W(I)/T
      T=1.
      V(N1)=T
      WRITE(3) (V(I),I=1,N1)
      IF(T.GT.EPS) GOTO 6
      L3=L3+1
  111 CONTINUE
   28 PRINT 118,K,EPS,T
      STOP
  118 FORMAT(13H W EQUAL ZERO,I5,2E15.5)
  123 FORMAT (4H-EIG,I4,2E20.10)
   13 FORMAT(1H0,I5,2E18.9)
   15 FORMAT(4(I4,E16.9))
   16 FORMAT (15I5)
   17 FORMAT(I4,2E16.9)
  130 FORMAT(1H-)
      END
      SUBROUTINE QR(N,SMACHEPS,E,D,IER)
      DIMENSION E(1),D(1)
C
C  THE SUBROUTINE QR COMPUTES THE EIGENVALUES BY MEAN OF QR METHOD FOR THE
C  THREE DIAGONAL MATRIX.
C
      IER=0
      DO 1000 I=2,N
 1000 E(I-1)=E(I)
      E(N)=0
      K=N-1
      DO 1001 L=1,N
      J=0
 1002 DO 1003 M=L,K
      IF(ABS(E(M)).LE.SMACHEPS*(ABS(D(M))+ABS(D(M+1)))) GOTO 1004
 1003 CONTINUE
      M=N
 1004 P=D(L)
      IF(M.EQ.L) GOTO 1001
      IF(J.EQ.30) GOTO 1005
```

```
      J=J+1
      G=(D(L+1)-P)/(2*E(L))
      R=SQRT(1.+G*G)
      Q=1.
      IF(G.LT.0.) Q=-1.
      G=D(M)-P+E(L)/(G+Q*R)
      S=1.
      C=1.
      P=D(M)
      MM=M-1
      DO 1006 II=L,MM
      I=M-1+L-II
      F=S*E(I)
      B=C*E(I)
      IF(ABS(F).LT.ABS(G)) GOTO 1007
      C=G/F
      R=SQRT(1.+C*C)
      E(I+1)=F*R
      S=1./R
      C=C/R
      GOTO 1008
1007  C=F/G
      R=SQRT(1.+C*C)
      E(I+1)=G*R
      S=C/R
      C=1./R
1008  F=C*D(I)-S*B
      G=C*B-S*P
      R=D(I)+P
      P=C*F-S*G
      G=S*F+C*G
      D(I+1)=R-P
1006  CONTINUE
      D(L)=P
      E(L)=G
      E(M)=0
      GOTO 1002
1001  CONTINUE
      DO 1009 I=1,N
      K=I
      P=D(I)
      III=I+1
      DO 1010 J=III,N
      IF(D(J).GE.P) GOTO 1011
      K=J
1010  P=D(J)
1011  IF(K.EQ.I) GOTO 1012
      D(K)=D(I)
      D(I)=P
1012  CONTINUE
1009  CONTINUE
      RETURN
1005  IER=1
      RETURN
      END
```

```
      SUBROUTINE AV(N,V,S)
      DIMENSION V(1),S(1)
      COMMON /10/ M(5000),A(5000),IS(5000)
C
C   THE SUBROUTINE AV COMPUTES THE PRODUCT S=A*V.
C
      DO 1 K=1,N
      S(K)=0
      K1=IS(K)
      K2=IS(K+1)-1
      DO 2 I=K1,K2
      J=M(I)
    2 S(K)=S(K)+A(I)*V(J)
      K3=K-1
      DO 3 I=1,K3
      K1=IS(I)
      K2=IS(I+1)-1
      DO 4 J=K1,K2
      IF(M(J).EQ.K) GOTO 5
    4 CONTINUE
    3 CONTINUE
      GOTO 1
    5 S(K)=S(K)+A(J)*V(I)
      GOTO 3
    1 CONTINUE
      RETURN
      END
      SUBROUTINE ORTSIM(N,K,W,S,C,EPS,T,L3,L4)
      DIMENSION W(1),S(1),C(1)
C
C   THE SUBROUTINE  ORTSIM PERFORMS THE  RE-ORTHOGONALIZATION OF THE MATRIX
C
      REWIND 3
      DO 1 L=1,K
      Z1=0
      N1=N+1
      READ(3) (S(J),J=1,N1)
      DO 2 I=1,N
    2 Z1=Z1+S(I)*W(I)
      Z2=Z1/S(N1)
      DO 3 I=1,N
    3 W(I)=W(I)-Z2*S(I)
    1 CONTINUE
      F=0
      DO 18 L=1,N
   18 F=F+W(L)*W(L)
      T=SQRT(F)
      C(K+1)=T
      PRINT 19,N,K,T,F,S(N1),Z1,Z2
   19 FORMAT(2I4,5E16.4)
      IF(T.GE.EPS) GOTO 7
      L4=-1
    7 CONTINUE
      DO 20 I=1,N
   20 W(I)=W(I)/T
      T=1
      W(N1)=T
      WRITE(3) (W(I),I=1,N1)
    9 FORMAT(3I5,3E15.5)
      RETURN
      END
```

Program 6 is for the computation of eigenvalues of a nonsymmetric sparse matrix /see Problem 6/.

The program reads the parameters N,R1,R2,EPS,EPS1 from the card in form (I5,4F15.8), where N is the order of matrix, EPS is the accuracy of eigenvalues, EPS1 investigates the singularity. The nonzero elements of the matrix and their column index are read from file 2 in form (4(I4,E16.9)). The nonzero elements in file 1 should be arrenged by rows, the program reads N+1 number from file 1 which points to the place of the first nonzero in each row in matrix. The end of nonzero elements is marked by a negative index.

The program executes the Lanczos method. The
SUBROUTINE AX (N,S,X) and
SUBROUTINE AY (N,S,Y) are used for computations of products S=AV and $S=A^TY$. The tridiagonal matrix is formed in array B(.) and C(.), /the third array consists of ones/. The
SUBROUTINE SEAT (N,R1,R2,I9,S,EPS) and
SUBROUTINE SAJS2 (N,P,NV) compute the eigenvalues of the tridiagonal matrix belonging to the interval (R1,R2) by means of the bisection method. The eigenvalues are formed in array B(.) in COMMON and they are written out.

The SUBROUTINE ORTOG performs the re-biorthogonalisation of the vectors.

If $|C_k| < EPS1$ in (2.7) the matrix decompose to parts. The program computs the eigenvalues of the first part, new biorthogonal vectors are computed and the computation is continued.

The program using 71 qp in core and 115 seg in disc is suitable for the computation of the eigenvalues of a matrix with N 500 and the number of nonzero elements 5000.

The list of the program:

```
C
C  THE PROGRAM COMPUTES THE REAL EIGENVALUES BELONG IN INTERVAL (R1,R2)
C  THE LANCZOS METHOD THEN THE BISECTION METHOD ARE USED.
C
C  THE INPUT OF PARAMETERS:
C  N   -  IS THE ORDER OF MATRIX
C  EPS  -  IS THE ACCURACY FOR EIGENVALUES
C
      DIMENSION IS(501),IO(5000),A(5000),B(500),C(500),(1(50
     10),X2(500),Y1(500),Y2(500),S(500)
      COMMON /1/ B,C /2/ IS,IO,A
      READ 1,N,R1,R2,EPS,EPS1
      PRINT 1,N,R1,R2,EPS,EPS1
      N1=N+1
      N2=N-1
      REWIND 1
      REWIND 2
      L3=2
      I1=1
      READ(1,9) (IS(I),I=1,N1)
      I2=IS(N+1)-1
C
C  THE NONZERO ELEMENTS AND THEIR COLUMN INDEXES ARE READ FROM FILE 2
C  THE ARRAY ELEMENT IS(I) LOOKS FOR THE FIRST NONZERO ELEMENT IN ROW I.
C
      READ(2,10) (IO(I),A(I),I=1,I2)
      PRINT 10,(IO(I),A(I),I=1,I2)
      DO 50 L=1,N
      DO 11 I=1,N
      Y1(I)=0.
      X1(I)=0.
      X2(I)=0
      Y2(I)=0
   11 CONTINUE
      X1(L)=1.
      Y1(L)=1.
      X1(N1)=1.
      WRITE(3) (X1(I),I=1,N1)
      WRITE(4) (Y1(I),I=1,N)
      C(1)=0
      S1=0
      DO 12 I=1,N
   12 S1=S1+X1(I)*Y1(I)
      DO 19 K=1,N
      CALL AX(N,S,X1)
      S2=0
      DO 13 I=1,N
   13 S2=S2+Y1(I)*S(I)
```

```
      PRINT 25,S1,S2
      B(K)=S2/S1
      DO 14 I=1,N
   14 X2(I)=S(I)-B(K)*X1(I)-C(K)*X2(I)
      CALL AY(N,S,Y1)
      DO 15 I=1,N
   15 Y2(I)=S(I)-B(K)*Y1(I)-C(K)*Y2(I)
      CALL ORTOG(N,K,X2,Y2,EPS1,M,S3,SX,SY,L3)
      I2=K+1
      IF(M.LT.0) C(K+1)=0
      IF(M.EQ.0) GOTO 60
      GOTO 51
   60 C(K+1)=S3*SY*SX/S1
  160 S1=S3
      DO 17 I=1,N
      E=X1(I)
      X1(I)=X2(I)
      X2(I)=E
      E=Y1(I)
      Y1(I)=Y2(I)
   17 Y2(I)=E
   19 CONTINUE
      IF(L.EQ.N) GOTO 2
      GOTO 51
    5 PRINT 10,L,S1,K
   51 CONTINUE
      I4=K-I1+1
      PRINT 25,(B(I),C(I),I=I1,I2)
      DO 18 I=1,N
   18 C(I)=C(I+1)
      DO 41 I=I1,I2
      I3=I-I1+1
      B(I3)=B(I)
      C(I3)=C(I)
   41 CONTINUE
      R3=R1
      R4=R2
      CALL SEAT (I4,R3,R4,NS,S,EPS)
C
C  THE EIGENVALUES ARE WRITTEN OUT
C
      DO 142 I=I1,I2
      I3=I2-I+1
      I5=I2-I+I1
  142 S(I5)=S(I3)
      I7=I1+NS-1
      PRINT 1,NS
      PRINT 22,(I,S(I),I=I1,I7)
      I1=I2
      IF(K.GE.N) GOTO 24
```

```
      DO 42 I=1,N
      XL(I)=0.
   42 YL(I)=0.
      GOTO 160
   20 PRINT 23,L,S1
   24 CONTINUE
   25 FORMAT(2E18.9)
    9 FORMAT (15I5)
   10 FORMAT(4(I4,E16.9))
   23 FORMAT(7H SZING:,I5,E15.5)
   22 FORMAT(4H-EIG,I5,E18.9)
    1 FORMAT(I5,4E15.8)
      END
      SUBROUTINE AX(N,S,X)
      DIMENSION S(1),X(1)
      COMMON /2/ IS(5000),IO(5000),A(50000)
C
C  THE SUBRPUTINE AX COMPUTES THE PRODUCT S=A*X.
C
      DO 1 I=1,N
      S(I)=0
      KK=IS(I)
      KV=IS(I+1)-1
      DO 1 K=KK,KV
      J=IO(K)
    1 S(I)=S(I)+A(K)*X(J)
      RETURN
      END
      SUBROUTINE AY(N,S,Y)
      DIMENSION S(1),Y(1)
      COMMON /2/ IS(5000),IO(50000),A(50000)
C
C  THE SUBROUTINE AY COMPUTES THE PRODUCT  S=Y*A
C
      DO 1 J=1,N
      S(J)=0
      DO 1 I=1,N
      IK=IS(I)
      IV=IS(I+1)-1
      DO 3 K=IK,IV
      IF(IO(K).EQ.J) GOTO 4
    3 CONTINUE
    1 CONTINUE
      GOTO 5
    4 S(J)=S(J)+A(K)*Y(I)
      GOTO 1
    5 RETURN
      END
```

```
      SUBROUTINE SEAT(N,R1,R2,I9,S,EPS)
      DIMENSION S(1)
C
C  THE SUBROUTINE SEAT COMPUTES THE BISECTION METHOD
C
      NS=1
      CALL SAJS2(N,R1,N1)
      CALL SAJS2(N,R2,N2)
      PRINT 13,N,N1,N2,R1,R2
      IF(N2-N1) 1,1,2
C
C  A MARC IS WRITTEN OUT IF EIGENVALUES ARE NOT IN INTERVAL (R1,R2)
C
    1 PRINT 3,R1,R2
      I9=0
      GOTO 10
    2 I9=N2-N1
C
C  IT IS WRITTEN OUT HOW MANY EIGENVALUES ARE IN INTERVAL (R1,R2)
C
      PRINT 4,I9,R1,R2
    5 CONTINUE
      IF(R2-R1-EPS.LE.0.) GOTO 20
      P=(R1+R2)/2
      CALL SAJS2(N,P,N3)
      IF(N2-N3) 7,7,8
    7 R2=P
      GOTO 5
    8 IF(N3-N1) 21,21,22
   21 R1=P
      GOTO 5
   22 R3=P
   23 P=(R3+R2)/2
      IF(R2-R3-EPS.LE.0.0) GOTO 20
      CALL SAJS2(N,P,N4)
      IF(N2-N4) 19,19,18
   19 R2=P
      GOTO 23
   18 R3=P
      GOTO 23
   20 S(NS)=P
      IF(NS.EQ.I9) GOTO 10
      NS=NS+1
      R2=P-2*EPS
      N2=N2-1
      GOTO 5
   10 CONTINUE
   13 FORMAT(3I5,2E16.6)
    3 FORMAT(23H IS NOT EIGENVALUE IN (,2E15.6,2H ))
    4 FORMAT(I5,21H EIGENVALUES ARE IN (,2E15.6,2H ))
      RETURN
      END
```

```
      SUBROUTINE SAJS2(N,P,NV)
      COMMON /1/ B(5  ),C(500)
      LOGICAL L
C
C  SUBROUTINE SAJS2 COMPUTES THE NUMBER OF CHANGE OF SIGN IN THE SEQUENCE
C  OF MINORS.
C
      NV=0
      L=.TRUE.
      D1=1.
      D2=B(1)-P
      IF(D2) 3,5,5
    3 L=.NOT.L
      NV=1
    5 CONTINUE
      DO 1 I=2,N
      D=D2*(B(I)-P)-C(I-1)*D1
      D1=D2
      D2=D
      IF(L.AND.D.GE.0.  .OR..NOT.L.AND.D.LT.0.0) GOTO 1
      NV=NV+1
      L=.NOT.L
    1 CONTINUE
    2 FORMAT(I5,2E16.7)
      RETURN
      END
      SUBROUTINE ORTOG(N,K,X2,Y2,EPS,M,S3,S1,S2,L3)
      DIMENSION X2(1),Y2(1),U(500),V(500)
C
C  THE SUBROUTINE ORTOG PERFORMS THE RE-BIORTHOGONALIZATION OF THE MATRIX
C
      REWIND 3
      REWIND 4
      M=
      DO 1 L=1,K
      Z1=
      Z2=0
      N1=N+1
      READ(3) (U(J),J=1,N1)
      READ(4) (V(J),J=1,N)
      DO 2 I=1,N
      Z1=Z1+V(I)*X2(I)
      Z2=Z2+U(I)*Y2(I)
    2 CONTINUE
      Z5=Z1/U(N1)
      Z6=Z2/U(N1)
      DO 3 I=1,N
      X2(I)=X2(I)-Z5*U(I)
    3 Y2(I)=Y2(I)-Z6*V(I)
    1 CONTINUE
      L2=L3
      DO 17 L=L2,N
      S1=
      S2=0
      DO 4 I=1,N
      S1=S1+X2(I)*X2(I)
    4 S2=S2+Y2(I)*Y2(I)
```

```
C
C     THE SUBROUTINE GENERATES A NEW BIORTOGONAL VECTORS
C     IF X2 OR Y2 IN NORM _ES THEN EPS.
C
      PRINT 99,N,K,S1,S2,U(N1),Z5,Z6
   99 FORMAT(2I5,5E12.4)
      IF(S1.GE.EPS.AND.S2.GE.EPS) GOTO 15
      IF(K.EQ.N) GOTO 97
      M=-1
      DO 6 I=1,N
    6 X2(I)=0
      X2(L)=1.
      REWIND 3
      REWIND 4
      DO 7 LL=1,K
      READ(3) (U(I),I=1,N1)
      READ(4) (V(I),I=1,N)
      Z1=0
      DO 8 I=1,N
      Z1=Z1+V(I)*X2(I)
    8 CONTINUE
      Z3=Z1/U(N1)
      DO 9 I=1,N
    9 X2(I)=X2(I)-Z3*V(I)
    7 CONTINUE
      DO 16 I=1,N
   16 Y2(I)=0.
      Y2(L)=1.
      REWIND 3
      REWIND 4
      DO 17 LL=1,K
      READ(3) (U(I),I=1,N1)
      READ(4) (V(I),I=1,N)
      Z1=0
      DO 18 I=1,N
      Z1=Z1+U(I)*Y2(I)
   18 CONTINUE
      Z3=Z1/U(N1)
      DO 19 I=1,N
   19 Y2(I)=Y2(I)-Z3*U(I)
   27 CONTINUE
      L3=L2+1
   17 CONTINUE
   15 CONTINUE
      DO 40 I=1,N
      X2(I)=X2(I)/S1
    4 Y2(I)=Y2(I)/S2
      S3=0
      DO 20 I=1,N
   20 S3=S3+X2(I)*Y2(I)
      IF(S3.LT.EPS) GOTO 27
      X2(N1)=S3
      WRITE(3) (X2(I),I=1,N1)
      WRITE(4) (Y2(I),I=1,N)
   97 CONTINUE
      RETURN
      END
```

References

[1] J.Gergely: Inversion of Large-Scale Sparse Matrices, Publicazioni Istituto per le applicationi del Calcolo Serie III.N.158,Róma, 1978.

[2] J.Gergely: Numerical methods for sparse matrices, treatise, Budapest, 1974. /in Hungarian/.

[3] J.K.Reid: FORTRAN subrutines for handling sparse linear programming bases, Oxfordshire, 1976.

[4] J.K.Reid: Sparse Matrices, York, 1976.

[5] Peters and J.H.Wilkinson: Eigenvalues of $Ax = \lambda Bx$ with band symmetric A and B, Com.J.12. 398-404.

[6] J.S.Duff: A survay of Sparse Matrix Research, Proceedings of the IEEE, vol.65, No.4.1977.

[7] A Dold and B.Echmann: Sparse Matrix Techniques, Springer, Copenhagen, 1976.

[8] B.M.Irons: A frontal solution program for finite element analysis, Int.J.Numer.Methods Eng., v.2. 5-32.

[9] R.P.Tewarson: Sparse Matrices, Academic Press, New York and London, 1973.

[10] J.H.Wilkinson: The Algebraic eigenvalue problem, Clarendon Press, Oxford, 1965.

99/1979 Ivics József: KGST Riga

100/1979 Téli iskola

1980-ban jelentek meg:

101/1980 Gerencsér László - Hangos Katalin:
Diszkrét lineáris sztochasztikus rendszerek
önhangoló szabályozása

102/1980 Pásztorné Varga Katalin: Rekurziv eljárás

103/1980 Gerencsér Piroska - Szép Endre - Zilahy Ferenc
Marton Zsolt: Robotmegfogók adaptivitása I.

104/1980 Knuth Előd - Radó Péter - Tóth Árpád:
Az SDLA előzetes ismertetése

105/1980 E. Knuth - P. Radó - Á. Tóth:
Preliminary description of SDLA

106/1980 Prékopa András: Sztochasztikus programozási
modellek és alkalmazásuk

107/1980 Kelle Péter: Megbizhatósági készletmodellek és
alkalmazásuk

108/1980 Almásy Gedeon: Mérlegegyenletek és mérési hibák

109/1980 Békéssy A. - Demetrovics J. - Gyepesi Gy.:
Relációs adatbázis logikai szintü vizsgálata
funkcionális függőségek szepontjából

110/1980 Gaál A. - Soltész J. - Ruda M. - Ratkó I.:
Tanulmányok a statisztikai adatfeldolgzásról

111/1980 Benedikt Szvetlána: Nem ismételhető döntéshozatal
analizise kockázattal járó esetekben

112/1980 Verebély Pál: Többprocesszoros, osztott intelligen-
ciáju grafikus rendszerek tervezési és megvalósitási
kérdései

113/1980	V. Visegrádi Téli Iskola
114/1980	Demetrovics János: Relációs adatmodell logikai és strukturális vizsgálata