Bulletin of the Chemists and Technologists of Macedonia, Vol. **14**, No. 2, pp. 125 – 136 (1995)

GHTMDD – 263
Received: January 20, 1995

ISSN 350 – 0136
UDC 681.3.06 Basic : 535.338.42

Computer application

CALCULATION OF THE MOMENTS OF INERTIA: A COMPUTER PROGRAM IN BASIC

Vladimir M. Petruševski[1] and **Ljupčo Pejov,**

Institut za hemija, Faculty of Natural Sciences, The "Sv. Kiril i Metodij" University, P.O.box 162, 91001 Skopje, Macedonia

In principle, the knowledge of molecular geometry allows one to calculate the principal values (eigenvalues) of the tensor of inertia. These quantities are essential in the calculation of the thermodynamic functions of molecules, for predictions of some details of their rotational spectra etc. Except for very simple molecules or molecules having high enough symmetry (at least C_{2v}) this problem is rather involved and requires the use of a computer. A program in BASIC, written by the authors, is presented in this paper. In addition, two worked examples are also given.

Key words: moments of inertia; rotations

INTRODUCTION

It is well known that motions of an N-atomic molecule may be described by 3N independent coordinates, corresponding to the 3N degrees of freedom. If the molecule is not linear, these 3N motions may be classified as 3 translations, 3 rotations and 3N-6 vibrations.

In the ideal gas – rigid rotor–- harmonic oscillator approximation, which is as a rule used in the calculation of thermodynamic functions [1], it is necessary to determine the set of normal vibrations, and the elementary moments of inertia[2] of the molecule in study. The corresponding information may be obtained by the study of the vibrational (IR and Raman) and microwave spectra of the compound in question.

Unfortunately, the data from rotational spectra and the numerical values of the moments of inertia are not always available. In certain cases, they can even not be obtained[3] in the usual way. However, if the geometry of the molecule is in some way known (e.g. as a result of a crystal structure determination; in the case of molecular crystals in which the interactions between the molecules are weak, one should expect only minor differences in the molecular geometry of the crystal and the gas phase), the moments of inertia may be calculated. The easiest way to do this is to use a computer. In certain cases (low molecular symmetry, large number of atoms) one is even forced to use a computer in the calculation of the moments of inertia, otherwise the calculation would be very lengthy and totally impractical. Let us discuss this in more detail.

The point is that the moments of inertia of molecules are described by a second rank tensor. The principal values of this tensor (the elementary moments of inertia) are what one really needs for most problems in physics/chemistry. These values may be calculated as the moments of inertia about the axes of, so to say, 'elementary rotations'. In certain cases these axes are determined by symmetry, and although lengthy, the calculation is not very involved. When this is not the case, one has to solve complicated secularequations *etc*. This is a typical eigenvalue problem, hence the use of a computer is strongly recommended.

THE COMPUTER PROGRAM

In order to solve the above problem, a computer program written in BASIC was developed[4]. The program was written on an IBM PC compatible microcomputer, using the QuickBasic interpreter/compiler

---

1 To whom correspondence should be addressed

2 By elementary moments of inertia are meant the moments of inertia about the axes of rotation, that are related to the semi-major axes of the ellipsoid of inertia.

3 This is the case with all centrosymmetric molecules, as they do not have pure rotational spectra; the same holds for all molecules with zero net dipole moment. Also, a molecule with at least C_2 axis will have one of its rotations "inactive" and, therefore, the value of one of the moments of inertia will remain unknown.

4 The program is available from the authors upon request.

programming package. There are no particular hardware requirements. Written in standard Microsoft BASIC, the program (if correctly retyped) should work with no modifications, on any other microcomputer equipped with a BASIC interpreter. Although not necessary, a mathematical co-processor is desirable, as it would significantly speed up the calculation.

Input data:

1) number of atoms in the molecule;

2) chemical symbols of the atoms;

3) Cartesian atomic coordinates, with respect to an arbitrary space-fixed rectangular coordinate system.

1) In the present version of the program the number of atoms is limited to 100. It may be easily increased by making a single change of the variable holding the maximum number of atoms (line 35 in the program list; see Appendix). In fact this number is limited only by the RAM of the computer.

2) The program contains a data-base of the elements in the periodic table (the chemical symbol and the relative atomic mass for each element have been entered in DATA lines). Every atom is identified by its symbol, while its mass is automatically assigned. If necessary, the data-base may be upgraded, e.g. to deal with isotopically labeled compounds (that is, one may add new data lines, containing information about the relative masses of important nuclides, such as ^{12}C; ^{13}C; ^{14}N; ^{15}N; ^{35}Cl; ^{37}Cl; ^{32}S; ^{34}S *etc.*).

3) The Cartesian coordinates of the atoms may be obtained from the fractional atomic coordinates (if the original data are taken from a crystal structure determination). A simple routine for conversion of fractional to Cartesian coordinates has been used in computer programs for calculation of distances and angles in crystal structures [2, 3]. Alternatively, if the distances and angles in the molecule are known, the Cartesian coordinates may be calculated using the program written by Eriksson [4] or any other similar program.

The subroutine for calculation of the eigenvalues and eigenvectors of a real, symmetric matrix was taken from the book by Mladenović *et al.* [5].

Computer output

The elementary moments of inertia (eigenvalues) are calculated and printed on the screen. The angles between the axes of elementary rotations (eigenvectors) and the coordinate axes are also calculated and printed. The angles between the rotational axes and an arbitrary line, passing through two atoms of the molecule, are calculated optionally.

Care should be taken when the moments of inertia are calculated for symmetric top and spherical top molecules. In this case two (*i.e.* all three) semi-major axes of the ellipsoid of inertia are equal and their orientation in space is quite arbitrary (providing they are mutually orthogonal). The calculated angles between these axes and the space-fixed coordinate axes, in such cases, have no physical significance at all.

It should be mentioned that the program may be useful for educational purposes, too. For more information, two worked examples are given below.

WORKED EXAMPLES

Example 1: HOH, DOD and HOD isotopomers of water.

HOH and DOD isotopomers of water have C_{2v} symmetry; the axes of elementary rotations are, therefore, fixed by symmetry (one of the axes is coincident with the C_2 symmetry axis; the second lies in the plane of the molecule and the third one is perpendicular to this plane). The symmetry of the HOD isotopomer is only *Cs* and the orientation of the first two axes has to be determined. The following geometry was used: R(O-H) = R(O-D) = 96 pm; ϕ(HOH) = ϕ(HOD) = ϕ(DOD) = 104.5°. The results are given in Table 1.

It is obvious that in all three cases one of the moments of inertia (I_3) is equal to the sum of the other two (that is $I_3 = I_1 + I_2$), which is a simple consequence of the planarity of the molecule. For HOH and DOD the orientation of the rotational axes is the same, but for HOD axes 1 and 2 are rotated about axis 3 for

Table 1

Moments of inertia for the isotopomers of water and angles between the axes of elementary rotations and the OH bond

Water isotopomer	$10^{47} \cdot I_1$/kg·m^2	$10^{47} \cdot I_2$/kg·m^2	$10^{47} \cdot I_3$/kg·m^2	Angle$_1$ / ° with O-H	Angle$_2$ / ° with O-H	Angle$_3$ / ° with O-H
HOH	1.929	1.027	2.956	52.25	37.75	90.00
DOD	3.846	1.843	5.689	52.25	37.75	90.00
HOD	1.216	3.078	4.293	58.81	31.19	90.00

approximately 6.5° (a consequence of the symmetry lowering). Also, for the HOD isotopomer, one rotation may be viewed as predominantly OH, and the other one as predominantly OD rotation. In line with this, the following relations hold:

$$I_1(\mathrm{HOH}) > I_1(\mathrm{HOD}) > I_2(\mathrm{HOH})$$

and

$$I_1(\mathrm{DOD}) > I_1(\mathrm{HOD}) > I_2(\mathrm{DOD})$$

These observations may help in understanding the somewhat peculiar behaviour of the twisting and wagging librations of water in crystallohydrates, when the spectra of all three isotopomers are studied [6, 7].

Example 2: the (hypothetical) $GeHF_2Cl$ molecule.

The authors are not aware whether such a molecule really exists; anyway, it may be dealt with as a hypothetical molecule in this example. All angles were assumed to be tetrahedral, and the bondlengths were calculated as sums of the covalent radii, i.e.:

R(Ge-H) = 154 pm

R(Ge-F) = 194 pm

R(Ge-Cl) = 221 pm

The use of the program written by Eriksson [4] gave certain values for the Cartesian atomic coordinates. These values were used as input data, and the following values for the moments of inertia were obtained :

$$I_1 = 4.532 \cdot 10^{-45}\ \mathrm{kg\ m^2}$$
$$I_2 = 1.896 \cdot 10^{-45}\ \mathrm{kg\ m^2}$$
$$I_3 = 3.261 \cdot 10^{-45}\ \mathrm{kg\ m^2}$$

The axis corresponding to I_2 (the smallest moment of inertia) makes an angle of 12.6° with the Ge-Cl bond, while those corresponding to I1 (the largest moment) makes an angle of 6.8° with the Ge-H bond, as one could expect intuitively.

LITERATURE

[1] G. Herzberg, *Molecular Spectra and Molecular Structure. II. Infrared and Raman Spectra of Polyatomic Molecules,* Van Nostrand, Princeton, 1956, pp. 501–530.

[2] V. M. Petruševski, *DISTAN (computer program for calculation of interatomic distances and angles in crystal structures),* unpublished.

[3] V. M. Petruševski, *CRYSCALC (computer program for various crystallographic calculations in crystal structures),* unpublished.

[4] A. Eriksson, *A computer program for conversion of internal to Cartesian coordinates,* unpublished.

[5] N. Mladenović, V. Spasić, M. Jovanović, *Numerički metodi za mikroračunare*, Tehnička knjiga, Beograd, 1986, pp. 118-120.

[6] A. Eriksson, J. Lindgren, *J. Mol. Struct.* **48**, 417 (1978).

[7] V. P. Tayal, B. K. Srivastava, D. P. Khandelwal, *Appl. Spectrosc. Rev.* **16**, 43 (1980).

Резиме

ПРЕСМЕТУВАЊЕ НА МОМЕНТИ НА ИНЕРЦИЈА: КОМПЈУТЕРСКА ПРОГРАМА ВО BASIC

Владимир М. Петрушевски и Љупчо Пејов

Институт за хемија, ПМФ, Универзитет "Св. Кирил и Методиј", 91001 Скопје, Македонија

Клучни зборови: моменти на инерција, ротации

Познавањето на молекуларната геометрија овозможува, во принцип, да се пресметаат главните компоненти (сопствените вредности) на тензорот на инерција. Овие величини се од клучно значење при пресметувањето на термодинамичките функции, при претскажувањето на одредени особености на ротационите спектри итн. Освен за многу едноставни молекули или за молекули со достатно висока симетрија (најмалку C_{2v}) ваквата задача е сложена и неопходно е во работата да се употреби компјутер. Во овој труд е прикажана компјутерска програма напишана во програмскиот јазик BASIC. Дадени се и резултатите од два решени примери.

APPENDIX
(Listing of the computer program)

```
DECLARE SUB Eigenvalue ()

REM ************************************************************************************
REM *                                                                                  *
REM *          This is a program for calculation of the                                *
REM *          moments of inertia for a given molecule                                 *
REM *                                                                                  *
REM *          Input data :                                                            *
REM *                                                                                  *
REM *          1. Chemical symbols for the atoms in the molecule                       *
REM *                                                                                  *
REM *          2. Atomic coordinates (Cartesian), with respect                         *
REM *             to an arbitrary coordinate system                                    *
REM *                                                                                  *
REM *          Authors: Ljupco Pejov & Vladimir Petrusevski,                           *
REM *          Institut za hemija, Prirodno-matematicki fakultet,                      *
REM *          Arhimedova 5, P.O.B. 162, 91001 Skopje, Makedonija                      *
REM *                                                                                  *
REM ************************************************************************************

CLS
DEFDBL A-Z              ' Double precision variables are used                        '

a$ = ""                 ' String variable used in WHILE...WEND structures            '
Prob$ = ""              ' ----------"----------                                       '

Aa = 6.022D+50          ' Renormalized Avogadro's constant = Na * 10^27              '
Counter = 0             ' Counter: atomic number of the atom in question             '
ErrorFlag = 0           ' Errorflag: if set => wrong atomic symbol                   '
m = 100                 ' Maximal number of atoms in the molecule                    '
Mr = 0                  ' Relative molecular mass of the compound                    '
N = 0                   ' Number of atoms in the molecule                            '
n1 = 0                  ' Ordinal number of an atom in the molecule                  '
n2 = 0                  ' ----------"----------                                      '
Pi = 4 * ATN(1#)        ' π - the circumference of the circle is 2rπ                 '
sumamx = 0              ' Variable used to calculate the centre-of-mass              '
sumamy = 0              ' ----------"----------                                      '
sumamz = 0              ' ----------"----------                                      '
vv = 0                  ' Length of a vector along a given line                      '
vx = 0                  ' Component of a unit vector along a given line              '
vy = 0                  ' ----------"----------                                      '
vz = 0                  ' ----------"----------                                      '
xc = 0                  ' Centre-of-mass coordinates of the molecule                 '
yc = 0                  ' ----------"----------                                      '
zc = 0                  ' ----------"----------                                      '
```

```
DIM Sym$(104)          ' Chemical symbols of the elements                    '
DIM Name$(m)           ' Chemical symbols of the atoms in the molecule       '

DIM Ar(m)              ' Relative masses of the atoms in the molecule        '
DIM AtMass(104)        ' Relative atomic masses of all elements              '
DIM c(3)               ' Cosines of the angles between a line                '
                       ' and the axes of elementary rotations                '
DIM Ev(3)              ' Eigenvalues of the tensor of inertia                '
DIM fi(3, 3)           ' Angles between the rotational and Cartesian axes    '
DIM Mass(m)            ' True mass of the atoms in the molecule              '
DIM T(3, 3)            ' Tensor of inertia of a given molecule               '
DIM theta(3)           ' Angles between a line and the rotational axes       '
DIM V(3, 3)            ' Eigenvectors of the tensor of inertia               '
DIM x(m)               ' Cartesian coordinates of the atoms in the molec.    '
DIM y(m)               ' ----------"----------                               '
DIM z(m)               ' ----------"----------                               '

Start:
CLS
PRINT : PRINT
PRINT "                MOMENTS  OF  INERTIA"
PRINT : PRINT
PRINT "     This is a program for calculation of the moments of in-          "
PRINT "     ertia, for a given N-atomic molecule (N 100). The in-            "
PRINT "     put data are the chemical symbols of the atoms building          "
PRINT "     the molecule and their Cartesian coordinates. The atomic         "
PRINT "     coordinates are given in picometers (pm).                        "
PRINT
PRINT "     The angles between the axes of elementary rotations and          "
PRINT "     the Cartesian (X,Y, Z) axes are also calculated. There is        "
PRINT "     an option for calculation of the angles between a line           "
PRINT "      (passing through two atoms of the molecule) and the axes        "
PRINT "     of elementary rotations                                         ."
PRINT : PRINT : PRINT
PRINT "                    C o n t i n u e (Y/N) ?                           "
  a$ = ""
  WHILE a$ <> "Y" AND a$ <> "N"
    a$ = UCASE$(INKEY$)
    IF a$ = "N" THEN GOTO EndOfProgram
  WEND
ReadData:              ' Read the data in the memory                         '
    FOR i = 1 TO 104
      READ Sym$(i):    ' Reads the chemical symbols of the elements          '
    NEXT i
    FOR i = 1 TO 104
      READ AtMass(i):  ' Reads relative atomic masses of the elements        '
    NEXT i
```

```
                    ' *** Input data 1. ***                                    '

  WHILE N < 2 OR N m > OR INT(N) < > N
    CLS
    PRINT : PRINT : PRINT
    INPUT       " Number of atoms in the molecule = "; N
  WEND

PRINT

PRINT "       Enter the chemical symbol for each atom :"
PRINT

    FOR i = 1 TO N          ' Cycle: identify the atom                         '

EnterSymbol:

      WHILE Name$(i) = "" OR ErrorFlag = 1
        PRINT "         Atom no. "; i; " = ";
        INPUT Name$(i)
        ErrorFlag = 0       ' Reset the ErrorFlag                              '
      WEND

AtomIdentification:

      Prob$ = Name$(i)      ' Try to identify i-th atom                        '

      Counter = 1           ' Counter: ordinal number of the atom '
                            ' in the periodic table                          '

      WHILE Prob$ < > "" AND Counter <=104
                                        ' IF Prob$ = Sym$(Counter) THEN        '

        IF Prob$ = Sym$(Counter) THEN   ' Atom identified;                     '
          Ar(i) = AtMass(Counter)       ' relevant data written in             '
          Mr = Mr + AtMass(Counter)     ' array Sym$( ) and Aa( )              '
          Prob$ = ""
        END IF
                                        ' ELSE:                                '

        Counter = Counter + 1           ' i-th atom was not identified;        '
                                        ' increase counter, until the          '
                                        ' atom is identified OR until          '
                                        ' the end of array is reached          '
    WEND
```

```
    IF Prob$ <> "" THEN                    ' i-th atom not identified;          '
                                           ' atomic symbol incorrect            '
      ErrorFlag = 1
      PRINT : PRINT "   Incorrect symbol of the atom."
      PRINT "  No element has a symbol ";    Prob$;   "!"
      PRINT "  Try again!"
      PRINT
      SLEEP 3
      GOTO EnterSymbol                     ' Enter symbol again                 '
    END IF

  NEXT i                                   ' Enter symbol of next atom          '


                        ' *** Input data 2. Atomic coordinates ***               '
PRINT : PRINT
PRINT "    Enter Cartesian atomic coordinates (/pm),"
PRINT

                        ' Data input & calculation of the Cartesian              '
                        ' coordinates of the centre-of-mass                      '
FOR i = 1 TO N
    PRINT
    PRINT "     x("; i; ") = ": INPUT x(i)
    PRINT "     y("; i; ") = ": INPUT y(i)
    PRINT "     z("; i; ") = ": INPUT z(i)
    sumamx = sumamx + Ar(i) * x(i)
    sumamy = sumamy + Ar(i) * y(i)
    sumamz = sumamz + Ar(i) * z(i)
NEXT i

xc = sumamx / Mr
yc = sumamy / Mr        ' xc, yc, zc – are the centre-of-mass coordinates        '
zc = sumamz / Mr

FOR i = 1 TO N
  xl(i) = (x(i) - xc)     ' Translation of the coordinate system;               '
  yl(i) = (y(i) - yc)     ' the origin is now in the centre-of-mass             '
  zl(i) = (z(i) - zc)
NEXT i

                        ' Define the tensor of inertia                          '
FOR i = 1 TO N
  T(1, 1) = T(1, 1) + Ar(i) * (yl(i) * yl(i) + zl(i) * zl(i))
  T(1, 2) = T(1, 2) – Ar(i) * xl(i) * yl(i)
  T(1, 3) = T(1, 3) – Ar(i) * xl(i) * zl(i)
  T(2, 1) = T(1, 2)
```

```
    T(2, 2) = T(2, 2) + Ar(i) * (xl(i) * xl(i) + zl(i) * zl(i))
    T(2, 3) = T(2, 3) – Ar(i) * yl(i) * zl(i)
    T(3, 1) = T(1, 3)
    T(3, 2) = T(2, 3)
    T(3, 3) = T(3, 3) + Ar(i) * (xl(i) * xl(i) + yl(i) * yl(i))
NEXT i

CALL Eigenvalue          ' Subprogram: Diagonalization of the tensor                    '
                         ' Ev(1) = I1 ; Ev(2) = I2 ; Ev(3) = I3                         '
FOR i = 1 TO 3
    Ev(i) = Ev(i) / Aa   ' Normalization: true numerical values (kg·m2)                 '
NEXT i

PRINT : PRINT

                         ' Eigenvalues of the tensor of inertia                         '

CLS
PRINT "Moments of inertia with respect to"
PRINT "the axes of elementary rotations :"
PRINT
PRINT USING "      \       \##.####^^^^\       \ "; "I1 ="; Ev(1); "kg·m2"
PRINT USING "      \       \##.####^^^^\       \"; "I2 ="; Ev(2); "kg·m2"
PRINT USING "      \       \##.####^^^^ \       \"; "I3 ="; Ev(3); "kg·m2"
PRINT : PRINT : PRINT

Angles:                  ' Angles : angle = ARCCOS(V(i,j))                              '

    FOR i = 1 TO 3
        FOR j = 1 TO 3

            IF V(i, j) = 0 THEN
                fi(i, j) = 90
            ELSE
                fi(i, j) = ATN(SQR(1-- V(i, j) ^ 2) / V(i, j)) * 180 / Pi
                ' IF fi(i, j) < 0 THEN fi(i, j) = 180 + fi(i, j)
            END IF

        NEXT j
    NEXT i

    PRINT "Angles / ° between axes of elementary rotations and coordinate axes :"
    PRINT
    PRINT USING "      \\          \\          \\      "; "X"; "Y"; "Z"
    PRINT
    PRINT "    I1";
    PRINT USING "  ###.###    ###.###  ###.###"; fi(1, 1); fi(1, 2); fi(1, 3)
```

```
    PRINT "       I2";
    PRINT USING "   ###.###     ###.###     ###.###"; fi(2, 1); fi(2, 2); fi(2, 3)
    PRINT "       I3";
    PRINT USING "   ###.###     ###.###     ###.###"; fi(3, 1); fi(3, 2); fi(3, 3)
    PRINT : PRINT : PRINT
Angles1:

    PRINT "Calculation of angles between the axes of rotation and"
    PRINT "a line passing through two atoms of the molecule (Y/N)?"
    PRINT : PRINT
    a$ = ""

    WHILE a$ < > "Y" AND a$ < > "N"
       a$ = UCASE$(INKEY$)
       IF a$ = "N" THEN GOTO EndOfProgram
    WEND

    CLS
    PRINT : PRINT
    INPUT "Ordinal number of the first atom = "; n1
    INPUT "Ordinal number of the second atom = "; n2
    PRINT : PRINT : PRINT

    vx = xl(n2) - xl(n1)
    vy = yl(n2) - yl(n1)                    ' Components of a vector along a line    '
    vz = zl(n2) - zl(n1)
    vv = SQR(vx * vx + vy * vy + vz * vz)   ' Length of this vector                  '

    vx = vx / vv
    vy = vy / vv                            ' Components of unit vector              '
    vz = vz / vv                            ' along the line                         '

    FOR i = 1 TO 3
       c(i) = V(i, 1) * vx + V(i, 2) * vy + V(i, 3) * vz

       IF c(i) = 0 THEN
          theta(i) = 90
       ELSE
          theta(i) = ATN(SQR(1-- c(i) ^ 2) / c(i)) * 180 / Pi
          ' IF theta(i) < 0 THEN theta(i) = 180 + theta(i)
    END IF

    NEXT i

    PRINT "The line passing through atoms "; n1; " and "; n2; "makes angles of"
    PRINT
```

```
    PRINT USING " ###.###    ###.###    ###.###";    theta(1); theta(2); theta(3)
    PRINT
    PRINT "degrees with the axes of elementary rotations."
    PRINT : PRINT : PRINT : PRINT
    SLEEP 4
    GOTO Angles1

EndOfProgram:
END

Symbols:                    ' Chemical symbols of the elements                    '

DATA H,He,Li,Be,B,C
DATA N,O,F,Ne,Na,Mg
DATA Al,Si,P,S,Cl,Ar
DATA K,Ca,Sc,Ti,V,Cr
DATA Mn,Fe,Co,Ni,Cu,Zn
DATA Ga,Ge,As,Se,Br,Kr
DATA Rb,Sr,Y,Zr,Nb,Mo
DATA Tc,Ru,Rh,Pd,Ag,Cd
DATA In,Sn,Sb,Te,I,Xe
DATA Cs,Ba,La,Ce,Pr,Nd
DATA Pm,Sm,Eu,Gd,Tb,Dy
DATA Ho,Er,Tm,Yb,Lu,Hf
DATA Ta,W,Re,Os,Ir,Pt
DATA Au,Hg,Tl,Pb,Bi,Po
DATA At,Rn,Fr,Ra,Ac,Th
DATA Pa,U,Np,Pu,Am,Cm
DATA Bk,Cf,Es,Fm,Md,No,Lw
DATA D

AtomicMasses:               ' Relative atomic masses of the elements              '

DATA 1.00797#,4.0026#,6.939#,9.0122#,10.811#,12.01115#
DATA 14.0067#,15.9994#,18.9984#,20.183#,22.9898#,24.312#
DATA 26.9815#,28.086#,30.9738#,32.064#,35.453#,39.948#
DATA 39.102#,40.08#,44.956#,47.90#,50.942#,51.996#
DATA 54.9381#,55.847#,58.9332#,58.71#,63.54#,65.37#
DATA 69.72#,72.59#,74.9216#,78.96#,79.909#,83.80#
DATA 85.47#,87.62#,88.905#,91.22#,92.906#,95.94#
DATA 97#,101.07#,102.905#,106.4#,107.87#,112.4#
DATA 114.82#,118.69#,121.75#,127.6#,126.9044#,131.3#
DATA 132.905#,137.34#,138.91#,140.12#,140.907#,144.24#
DATA 145#,150.35#,151.96#,157.25#,158.924#,162.5#
DATA 164.93#,167.26#,168.934#,173.04#,174.97#,178.49#
DATA 180.948#,183.85#,186.2#,190.2#,192.2#,195.09#
DATA 196.967#,200.59#,204.37#,207.19#,208.98#,210#
```

```
DATA 210#,222#,223#,226#,227#,232.038#
DATA 231#,238.03#,237#,242#,243#,247#
DATA 247#,249#,254#,253#,256#,254#,257#
DATA 2.01#

SUB Eigenvalue STATIC      ' Modified from N. Mladenovic, V. Spasic and M.
                           ' Jovanovic "Numerical methods for microcomputers",
                           ' Tehnicka knjiga, Beograd, 1986, pp. 118–120.
SHARED T(), Ev( ), V( )

N = 3: e = .000001#: eee = 1D–22

FOR i = 1 TO N
      FOR j = 1 TO i
         a(i * (i–- 1) / 2 + j) = T(i, j) + eee
      NEXT j
NEXT i

REM *** initialization ***

FOR i = 1 TO N
      FOR j = 1 TO N
         V(i, j) = 0
      NEXT j
      V(i, i) = 1
NEXT i

REM *** normalization ***

p = 0

FOR i = 2 TO N
      FOR j = 1 TO i–- 1
         a = a(i * (i – 1) / 2 + j): p = p + a * a
      NEXT j
NEXT i

p = SQR(2 * p): gr = e / N * p

Norma:
p = p / N


REM *** matrix transform ***

Trans:
l = 0
```

```
FOR j = 2 TO N
    jj = j * (j - 1) / 2 + j
    FOR i = 1 TO j - 1
      ii = i * (i-- 1) / 2 + i: ji = jj - j + i
      aii = a(ii): aji = a(ji): ajj = a(jj)
      IF ABS(aji) < p THEN GOTO Endloop
      l = 1: b = (aii - ajj) / 2
      IF b = 0 THEN a =--1: GOTO a
      a = -SGN(b) * aji / SQR(aji * aji + b * b)
a:    s = a / SQR(2 * (1 + SQR(1 - a * a)))
    c = SQR(1 - s * s)
    FOR k = 1 TO N
      kk = k * (k -- 1) / 2 + k: ki = kk - k + i
      IF k > i THEN GOTO b
      ki = ii-- i + k

b:    kj = kk-- k + j
      IF k > j THEN GOTO c
      kj = jj-- j + k

c:    V = a(ki)
      IF k = j THEN GOTO d
      a(ki) = V * c-- a(kj) * s

d:    a(kj) = V * s + a(kj) * c
      V = V(i, k): a = V(j, k): V(i, k) = V * c-- a * s
      V(j, k) = V * s + a * c
    NEXT k

    a = c * c: b = s * s: V = 2 * aji * c * s
    a(ii) = aii * a + ajj * b - V
    a(jj) = aii * b + ajj * a + V: a(ji) = 0
Endloop:
    NEXT i
NEXT j

REM *** check-out ***

IF l = 1 THEN GOTO Trans
IF p > gr THEN p = p / N: GOTO Norma

REM *** output results ***

FOR k = 1 TO N
    Ev(k) = a(k * (k-- 1) / 2 + k)
NEXT k
END SUB
```