Pakistan J. Sci. Ind. Res., Vol. 30, No. 5, May 1987

HOME COMPUTER IN MOLECULAR ORBITAL CALCULATIONS
Part I. The Huckel Method

M. Mohammad A.Y. Khan and Romana Qureshi

Electrochemistry Group, Quaid-i-Azam University, Islamabad

(Received September 23, 1986; revised March 16, 1987)

Computer Programs in BASIC language for molecular orbital calculations have been developed for home computer (10 K memory). The M.O. Program can be used for research as well as teaching purposes. The programs are simple and tractable and can be used by nonprofessionals with ease. The program has been tested and found working with less than 15 seconds' time taken for allyl system.

Key words: Home computer; M.O. program Huckel Method; Basic program.

INTRODUCTION

The importance of M.O. calculations and utilization of various parameters (orbital energies, electron densities etc.) in teaching and research is well established. With the advent of computers, such calculations have become a lot easier. However, even with the terminal facilities available, the big computers are out of reach for many. Besides, because of time sharing, running M.O. programs for teaching purposes could be quite expensive. Also some of the programs written by professional programmers, though available in the library, cannot be easily modified to suit our (and the computer's) requirements. If these M.O. calculations can be carried out on small inexpensive and easily accessible computers, many of the above problems can be solved.

With the advent of Home Computers which are now easily accessible even to high schools (and to individuals), it is lot easier and more convenient to handle M.O. calculations by these computers. Besides, because of BASIC Language, the user can understand more easily about the techniques of M.O. calculations. It is more easily traceable to the user.

A computer program in BASIC language has been developed to carry out Huckel M.O. calculation. The program has been tested on a Home Computer with 10K memory.

Molecular orbital theory. Here only a summary of the theory is given. For details one is referred to the literature [1].

Each molecular orbital can be expressed as a linear combination of n atomic orbitals (where n is the number of basis orbitals):

$$\Psi_j = \sum_i^n C_{ji}\,\phi_i \qquad (1)$$

C_{jr} = coefficient of the r^{th} atomic orbital in j^{th} M.O.
ϕ_i = A.O. for the r^{th} atom

The Huckel Molecular Orbital method is [2] the simplest method for calculating the energies and coefficients of the molecular orbitals.

Using the variational method, a set of linear equations is obtained:

$$\Sigma\, C_j (H_{ij} - S_{ij}\, E) = O;\ i = 1, 2, . . n \qquad (2)$$

Where the quantities

$$H_{ij} = H_{ji} = \int \phi_i^* \, H \, \phi_j \, d\tau \qquad (3)$$

$$S_{ij} = \int \phi_i^* \, \phi_j \, d\tau \qquad (4)$$

are numbers which can be computed from the starting ϕ_i and known H. In the matrix form this set of n equations can be represented as [3].

$$HC = SCE. \qquad (5)$$

C = matrix of coefficients (or matrix of eigenvectors). Each column of the matrix C refers to different M.O. elements; S_{ij} of the matrix S can be represented as

$$S_{ij} = \int \phi_i^* \, \phi_j \, d\nu$$

Under the assumption that the basis function is orthonormal and matrix S becomes equal to the identity matrix and we are left with the equations $C^+ HC = CE$. Also Matrix elements of H are known then C and E can be calculated. E in this case is a diagnonal matrix.

The diagonal elements of E correspond to the eigenvalues or energies of the M.O.'s.

The problem is now that given a matrix H, to find unitary matrix C such that C^+ HC is diagonal. Details of matrix diagonalization can be obtained from ref. 3. and ref. 4.

The next problem is the evaluation of the matrix elements H_{ij}.H_{ii} represents the energy of an electron in the i^{th} atomic orbital. This integral known as coulumb integral is denoted by the symbol α (= $\int \phi_i H \phi_i \, d\nu$) where ϕ is the $2p_z$ A.O. of a carbon atom [2]. H_{ij} represents the energy of interaction of orbitals and as a simplification if atoms i and j are bonded then $H_{ij} = \beta$, otherwise (if i and j are not bonded) $H_{ij} = O$. This is known as the resonance integral. This is the case when we are considering simple conjugated or aromatic hydrocarbons. However in a more general form the matrix elements of the Hamiltonian can be represented as $H_{ii} = \alpha + h_x\beta$ and for neighbouring atoms $H_{ij} = \beta = k_c \beta_o$ where β_o is taken to be a unit of energy.

All roots are computed relative to α, which is usually taken for convenience to be zero. According to convention $h_x = O$ if the i^{th} A.O. is the $2P_z$ A.O. However if instead of carbon we have some other heteroatom X as part of the system, then h_x may have a positive or negative value depending upon the electronegativity of the heteroatom relative to the carbon atom.

For a C-X bond the value of the parameter $k_{C\text{-}X}$ is not equal to unity [5]. Once the energies and co-efficients of the molecular orbitals are calculated then the electrons are assigned to the M.O.'s according to Pauli's exclusion principle.

According to Coulsion the partial mobile bond order of the r-s bond in the j^{th} M.O. is defined as [2].

$$P^j_{rs} = C_{jr} C_{js} \tag{6}$$

$$P_{rs} = \Sigma P^j_{rs} = \Sigma n_j C_{jr} C_{jr} \tag{7}$$

Bond orders may be associated with the binding power of a bond. The electron density at an atom, q_r, is the sum of electron density contributed by each electron in each M.O.

$$q_r = \Sigma C^2_{jr} \tag{8}$$

C_{jr} is the coefficient of atom r in the j^{th} M.O., which is occupied by n_j electrons. The sum is taken over all the M.O.'s.

Huckel calculations are semiempiricial in nature and a number of approximations have been introduced. Some of the approximations can be reduced by further techniques within the framework of H.M.O. method; w-technique [2,6] is one of these techniques.

Computational techniques. Molecular Orbital Calculations have been carried out on a 10K Home Computer Sharp (PC-1500) using the simple Huckel Molecular Orbital method. The method is based upon matrix diagonalization in BASIC language and is designed to provide processing for one molecule with one electronic configuration at a time.

A BASIC program for solving Huckel-electron problem is written in the form of subprograms, each of which performs a specific part of the complete calculation.

It consists of a *Main Control Program* which controls the sequence of operation, and several subroutines. These subroutines being INPUT, DIAGONALIZATION, ORDER, BOND ORDER, OUTPUT, PRINTOUT INPUT DATA, PRINTOUT RESULTS, OUTPUT.

The functions of the above mentioned SUBROUTINES are generally known and these do not have to be elaborated briefly:

INPUT Subroutine reads the number of conjugated atoms, the Huckel matrix and the number of electrons. It reads only the upper semi matrix (of the Huckel matrix) and brings down the upper semi matrix to the lower semimatrix.

The DIAGONALIZATION subroutine diagonalizes the matrix by the Jacobi method.

The subroutine ORDER puts the eigen-values and eigenvectors in a sequence from the lowest to the highest. Negative eigen-values correspond to *bonding* energy levels. The nond order subroutine calculate the charge densities and bond order (eqns 7 and 8). The function of other subroutines are obvious.

There are two points worth mentioning, viz. (i) Form of the Input matrix and (ii) Timing.

(i) Form of the input matrix. The input matrix H (I, J) contains O'S and 1'S for straight/conjugated aromatic hydrocarbon. For hydrocarbon, the diagonal elements H (I, I) are set equal to zero, off diagonal elements are 1'S for neighboring atoms and O's for non-neighbouring.

In heteroatomic system not all diagonal elements are O. If the heteroatom is at the Ith position, the H (I, I) is equal to hx. Also H (I, J) is equal to $k_{c\text{-}x}$ when J is the neighbour of the heteroatom.

A point to remember is that as the energy levels are arranged in ascending order (from -ve to +ve values) electron should be filled accordingly.

As an example consider the case of pyridine. Questions will appear on the screen for the number of atoms, H (I, J) matrix and the number of electrons. These are to be "Entered".

```
H (1, 1) = ? 0.5     H (2, 2) = ? 0     H (4, 4) = ? 0
H (1, 2) = ? 0.8     H (2, 3) = ? 1     H (4, 5) = ? 1
H (1, 3) = ? 0       H (2, 4) = ? 0     H (4, 6) = ? 0
H (1, 4) = ? 0       H (2, 5) = ? 0
H (1, 5) = ? 0
H (1, 6) = ? 0.8
                     H (3, 3) = ? 0
                     H (3, 4) = ? 1     H (5, 5) = ? 0
NE (1) = ? 2         H (3, 5) = ? 0     H (4, 6) = ? 1
NE (2) = ? 2         H (3, 6) = ? 0     H (4, 6) = ? 0
NE (3) = ? 2
NE (4) = ? 0
NE (5) = ? 0
NE (6) = ? 0
```

(ii) Timing.

The time taken by the computer to run the above mentioned programs for allyl radical (or cation) and benzene molecule (or ion) is given below

Alkyl radical	18 secs
Benzene	100 secs

on larger computer, e.g. APPLE IIC, considerable less time is required for computation.

In case of the Jacobi diagonalization time taken is proportional to N_3 (Where N = No. of atoms in a molecule)

$$\frac{T_1}{T_2} = \frac{N_1^3}{N_2^3}$$

The above mentioned data are found to be in rough agreement with this rule.

Printouts: Input Data and Results (H.M.O.)

```
              H.M.O CALCULATIONS

 ALLYL RADICAL               H (2, 3) =–1
                             H (3, 3) =  0

 INPUT DATA                  NE (1) = NO. OF ELECTO
 H (1, 1) =  0               NS IN THE ITH M.O
 H (1, 2) =  1               NE (1) = 2
 H (1, 3) =  0               NE (2) = 1
 H (2, 2) =  0               NE (3) = 0

                   RESULTS

EIGENVALUES                  CHARGE DENSITIES
E (1) =– 1.414213563         Q (1) = 1
E (2) =– 2.850934661         Q (2) = 1
E – 12                       Q (3) = 9.999999998
E (3) = 1.414213563           E– 01
```

```
EIGENVECTORS                 812E – 01
U (1, 1) =  5.000000         U (3, 3) =– 0.5
001E – 01
U (1, 2) =– 7.071067         BONDORDERS
811E – 01                    P (1, 1) = 1
U (1, 3) =– 5.000000         P (1, 2) = 7.071067
001E – 01                    812E – 01
U (2, 1) = 7.071067          P (1, 3) = 5.2E – 11
812E – –01                   P (2, 1) = 0
U (2, 2) = 2.099388          P (2, 2) = 1
572E – 10                    P (2, 3) = 7.071067
U (2, 3) = 7.071067          812E – 01
812E – 01                    P (3, 1) = 0
U (3, 1) = 4.999999          P (3, 2) = 0
999E – 01                    P (3, 3) = 9.999999
U (3, 2) = 7.071067          998E – 01
```

1. H.M.O. Programme For Sharp PC – 1500 Home Computer (10 K)

```
    H.M.O.
 10: Print "H.M.O. Calculations"
 15: RC = 1
 20: GOSUB 80
 25: GOSUB 230
 30: GOSUB 600
 35: GOSUB 700
 40: GOSUB 800
 45: END

Bondorders Subroutine
700: Rem Subroutine Bondorders
705: FOR R = I TO N
710: FOR S = R TO N
715: SM = 0
720: FOR J = I TO N
725: SM = SM ÷ NE (J) *U (R, J) * U (S, J)
730: NEXT J
740: P (R, S) = SM
745: NEXT S
750: NEXT R
755: RETURN
Input Subroutine
 80: REM "INPUT SUB ROUTINE"
 85: INPUT "NAME?"; N$
 90: INPUT "NO OF AT OMS = ?"; N
 95: DIM H (N, N): DIM HD (N): DIM NE (N): DIM U
     (N, N): DIM UT (N): DIM P (N, N)
100: WAIT 0
105: FOR I = I TO N
```

```
110: FOR J = I TO N
115: CLS: PRINT "H ("; I; ", "; J;") =";
120: INPUT H (I, J)
125: H (J, I) = H (I, J)
130: NEXT J
135: NEXT I
140: FOR I = I TO N
145: CLS: PRINT "NE ("; I;") =";
150: INPUT NE (I)
155: NEXT I
160: WAIT 100: CLS: PRINT "CHECK T HE INPUT H
     (I, J)"
165: FOR I = I TO N
170: FOR J = I TO N
175: CLS: WAIT 0: PRINT "H ("; I;" "; J; ") = "; H (I, J)
180: CURSOR 20: INPUT H (I, J)
185: H (I, J) = – H (I, J)
190: H (J, I) = H (I, J)
195: HD (I) = H (I, I)
200: NEXT J
205: NEXT I
210: WAIT
215: CLS: INPUT "PR INTOUT INPUT D ATA REOD?";
     U$
220: IF U$ = "YES" THEN GOSUB 400 0
225: RETURN
```

Diagonalization Subroutine

```
230: REM DIAGONALIZATION
235: REM SUBROUTINE
240: FOR J = ITO N
245: FOR I = ITO N
250: U (I, J) = 0
255: U (I, I) = 1
260: NEXT I
265: NEXT J
270: EP = IE – 16
275: MX = 0
280: FOR I = 2 TO N
285: FOR J = I TO (1 – 1)
298: H (I, I) @ HD (I)
295: H (J, J) = HD (J)
300: SQ = H (I, J) 2
310: IF SQ > MXTHEN LET MX = SQ
315: IF SQ < = EPTHEN GOTO 515
320: D = HD (I) – HD (J)
325: IF D > = 0 THEN GOTO 340
330: SN = – 2: D = – D
335: GOTO 345
340: SN = 2
345: TD = (D + √ ( ( D * D) + (4 * SQ) ) )
350: TN = (SN * H (I, J) )/TD
355: C = I/(√ (I ÷ (TN * T N) ) )
380: S = C * TN
370: FOR K = I TO N
380: XJ = C * U IK, J) – S * U (K, I)
385: U (K, I) = S * U (K, J) ÷ C * U (K, I)
390: U (K, J) = XJ
395: IF K = J THEN GOTO 485
400: IF K > J THEN GOTO 435
410: XT = C * H (J, K) – S * H (I, K)
420: H (I, K) = S * H (J, K) + C * H (J, K)
425: H (J, K) = XJ
430: GOTO 485
435: IF K = I THEN GOTO 485
440: IF K > I THEN GOTO 470
450: XJ = C * H (K, J) – S * H (I, K)
455: H (I, K) = S * H (K, J) + C * H (I, K)
460: H (K, J) = XJ
465: GOTO 485
470: XJ = C * H (K, J) – S * H (K, J)
475: H (K, I) = S * H (K, J) + C * K (K, I)
480: H (K, J) = XJ
495: NEXT K
490: HD (J) = (S * S * H (J, I) ) + (S * S * H (J, J) ) +
     (2 * S * S * H (I, J) )
500: HD (J) = (S * S * H (J, J) ) + (S * S * H (J, I) ) –
     (2 * S * S * H (J, J) )
510: H (J, J) = 0
515: NEXT J
520: NEXT I
525: IF MX > EP THEN GOTO 275
530: RETURN
540: END
```

Output Sueroutines

```
800: REM SUBROUTINE OUTPUT
805: BEEF 5, 10, 100
810: PRINT "EIGENUA LUES"
815: FOR I = I TO N
820: WAIT
825: PRINT "E ("; I;' ") HD (I)
830: NEXT I
840: INPUT "EIGENUE GTORS REQD. ?"; U$
845: IF U$ = "NO" THEN GOTO 875
850: FOR J = I TO N
860: PRINT "U ("; I; " "; J.; ") = "; U (I, J)
865: NEXT I
870: NEXT J
875: INPUT "CHARGE DENSITIES REQD ?"; C$
```

```
880: IF C$ = "NO" THEN GOTO 900
885: FOR I = I TO N
890: PRINT "Q ("; I; ") = "; P (I, I)
895: NEXT I
900: INPUT "BOUNDORD ERS REQD? "; P$
905: IF P$ = "NO" THEN GOTO 935
910: FOR I = I TO (N – I)
915: FOR J = (I + I) TO N
920: PRINT "P ("; I;" "; J; ") = "; P (I, J)
925: NEXT J
930: NEXT I
935: INPUT "PRINTOUT RESULTS REQD?"; U$
940: IF U$ = "TES" THEN GOSUB 800 0
950: RETUN
```

Order Subroutine

```
600: REM SUBROUTINE ORDER
605: FOR K = I TO N
610: HT = HD (K)
615: JT = K
620: FOR J = K TO N
625: IF HD (J) > = HT THEN GOTO 640
630: HT = HD (J)
635: JT = J
640: NEXT J
645: HD (JT) = HD (K)
650: HD (K) = HT
655: FOR I = I TO N
660: UT (I) = U (I, JT)
665: U (I, JT) = U (I, K)
678: U (I, K) = UT (I)
675: NEXT I
680: NEXT K
685: RETURN
690: END
```

Printout Subroutines

```
4000: REM SUBROUTINE
4001: REM PRINTOUT INPUT DATA
4005: CSIZE 3: LF 3
4010: ON RC GOTO 4020, 34030, 4040
4020: L PRINT "H. M. O CALCULATIONS"
4025: GOTO 4050
4030: L PRINT "H.M.O WITH W-TECHNIQUE"
4035: GOTO 4050
4040: L PRINT "H.M.O WITH MCLAC HLAN METHOD"
4050: LF I: C SIZE 2
4860: L PRINT N$
4070: LF 1
4080: L PRINT "INPUT DATA"
5000: FOR I = I TO N
5010: FOR J = I TO N
5020: L PRINT "H ("; I;", "; J; ") = "; H (I, J)
5030: NEXT J
5040: NEXT I
5060: LF 2
5070: L PRINT "NE (I) = NO. OF ELECTONS IN THE
      ITH M.O"
5080: FOR I I TO N
5090: L PRINT "NE ("; I; ") = "; NE (I)
6000: NEXT I
6010: RETURN
6050: REM PRINTOUT INPUT
6055: REM DATA S.T.C.F.
6060: LT I: C SIZE 3
6070: L PRINT "S.C.C CALCULATIONS"
6080: C SIZE 2: LT I
6090: L PRINT N$
7000: LF 1
7010: FOR I = I TO N
7020: L PRINT "X ("; I; ") = "; X (I); " "; "Y ("; I; ") =
      "; Y (I)
7040: NEXT I
7045: LF I
7050: L PRINT "NE (I) = NO. OF ELECTRONS IN THE
      ITH M.O.
7060: FOR I = I TO N
7070: L PRINT "NE ("; I; ") = "; NE (I)
7080: NEXT I
7090: RETURN
8000: REM SUBROUTINE PRINTOUT RESULTS
8005: LF 2: C SIZE 3
8010: L PRINT "RESULTS"
8022: IF RC = I THEN GOTO 8040
```

Printout Results Subroutine

```
8000: REM SUBROUTINE PRINTOUT RESULTS
8005: LF 2: C SIZE 3
8010: L PRINT "RESULTS"
8020: IF RC = I THEN GOTO 8040
8030: C SIZE 2: L PRINT "ITERATION NO. "; K, K
8040: C SIZE 2: LF 1
8050: L PRINT "EIGEN VALUES"
8060: L PRINT "E ("; I; ") = "; HD (I)
8080: NEXT 1
8090: LF 2: L PRINT "CHARGE DENSITIES"
9000: FOR I = I TO N
9010: L PRINT "Q ("; I; ") = "; P (I, I)
9020: NEXT 1
9025: LF 2
9030: L PRINT "EIGENVECTORS"
```

```
 9040:  FOR I = I TO N
 9050:  FOR J = I TO N
 9060:  L PRINT “U (”; I; ”, ”; J; ”) = ”; U (I, J)
 9070:  NEXT J
 9080:  LF 2
 9090:  L PRINT “BOND ORDERS”
10000:  FOR I = I TO N
10010:  FOR J = I TO N
10020:  L PRINT “P (”; I; ”; ”; J; ”) = ”; P (I, J)
10030:  NEXT J
10040:  NEXT I
10050:  RETURN
```

2. H.M.O. Programme for Sharp PC – 7000; Apple II.

```
  1   TGOTO 2600
 10   PRINT “H.M.O. Calculations”
 15   RC = 1
 20   GOSUB 80
 25   GOSUB 230
 30   GOSUB 600
 35   GOSUB 700
 40   RETURN
 45   END
 79   REM
 80   REM “input subroutine”
 81   REM
 85   INPUT “name”; N$
 90   INPUT “no. of atoms”; N
 95   DIM H (N, N): DIM HD (N): DIM NE (N): DIM U
      (N, N): DIM UT (N): DIM P (N, N)
105   FOR I = I TO N
110   FOR J = I TO N
115   FOR J = I TO N
115   PRINT “H (”; I; ”, ”; J; ”) = ”;
120   INPUT H (I, J)
125   H (J, I) = H (I, J)
130   NEXT J
135   NEXT I
140   FOR I = I TO N
145   PRINT “NE (”; I; ”) = ”;
150   INPUT NE (I)
155   NEXT I
159   PRINT
160   COLOR 3: PRINT “VERIFICATION of H (I, J)”:
      PRINT “To confirm, press the SPACE bar”
161   PRINT “To alter, press the = key, then key-in the
      correct value, then press CR”: COLOR 7
165   FOR I = I TO N
170   FOR J = I TO N
175   PRINT “H (”; I; ”; ”:; J; ”) = ”, H (I, J),
177   U$ = IN KEY $: IF U$ = ” ” THEN 177
178   IF U$ = ” ” THEN PRINT “O.K.”: GOTO 185
179   IF U$ = ” = ” THEN PRINT “ = ”; : INPUT H (I, J):
      GOTO 185
185   H (I, J) = – H (I, J): H (J, I) = H (I, J): HD (I) = H
      (I, I)
200   NEXT J
205   NEXT I
215   PRINT “print-out of input data required? (y/n)”,
220   U$ = IN KEY$: IF U$ = ” ” THEN 220
221   PRINT U$
222   PRINT “BUSY”
225   RETURN
229   REM
230   REM diagonalisation subroutine
231   REM
240   FOR J = I TO N
245   FOR I = I TO N
250   U (I, J) = O: U (I, I) = I
260   NEXT I
265   NEXT J
270   EP = IE – 16
275   MX = O
280   FOR I = 2 TO N
285   FOR J = I TO (I – I)
290   H (I, I) = HD (I): H (J, J) = HD (J): SQ = (H (J, J) )
      2
310   IF SQ > MX THEN LET MX = SQ
315   IF SQ <= EP THEN 510
320   D = HD (I) – HD (J)
325   IF D >= O THEN 340
330   SN = – 2: D = – D
335   GTO 345
340   SN = 2
345   TD = D + SQR ( (D * D) + (4 * SQ) ): TN = (SN * H
      (I, J) )/TD
355   C = I/(SQR (1 + (TN * TN) ) ): S = C * TN
370   FOR K = I TO N
380   IF K = J THEN 485
400   IF K > J THEN 435
450   XJ = C * H (K, J) – S * H (I, K): H (I, K) = S * H
      (K, J) + C * H (I, K): H (K, J) = XJ
465   GOTO 485
470   XJ = C * H (K, J) – S * H (K, I): H (K, I) = S * H
      (K, J) + C * H (K, I): H (K, J) = XJ
485   NEXT
490   HD (I) = (C * C * H (I, I) ) + (S * S * H (J, J) ) +
      ( 2 * S * C * H (I, J) )
500   HD (J) = (C * C * H (J, J) ) + (S * S * H (I, I) ) –
```

```
      (2 * S * C * H (I, J) )
505   H (I, J) = O
510   IF MX > EP THEN 275
530   RETURN
540   END
599   REM
600   REM order subroutine
601   REM
605   FOR K = I TO N
615   HT = HD (K): JT = K
620   FOR J = K TO N
625   IF HD (J) > = HT THEN 640
630   HT = HD (J): JT = J
640   NEXT J
645   HD (JT) = HD (K): HD (K) = HT
655   FOR I = I TO N
660   UT (I) = U (I, JT): U (I, JT) = U (I.K) = UT (I)
670   NEXT: NEXT
685   RETURN
690   END
699   REM
700   REM subroutine bond-orders
701   REM
705   FOR R = I TO N
710   FOR S = R TO N
715   SM = O
720   FOR J = I TO N
725   SM = SM + NE (J) * U (R, J) * U (S, J)
730   NEXT J
740   P (R, S) = SM
745   NEXT S
750   NEXT R
755   RETURN
799   REM
800   REM
801   REM output subroutine
2600  REM bondorders and forceconstants
2605  GOSUB 10
2610  FOR R = I TO (N – I)
2615  FOR S = (R + I) TO N
2620  IFH (R, S) = O THEN LET P (R, S) = O
2625  P (S, R) = P (R, S)
2645  NEXT: NEXT
2655  IF ( (N/2) – INT (N/2) ) = OTHEN GO TO 2685
2660  X = (N + 1)/2
2675  NE (X) = 2
2680  GOTO 2700
2685  X = (N/2) + 1
2695  NE (X) = NE (X) + 1
2700  GOSUB 700
2750  L = O
2710  FOR R = I TO (N – I)
2715  FOR S = (R + I) TO N
2720  IF H (R, S) = O THEN LET P (R, S) = O
2730  K = 5.47 * (1 + P (S, R) ) – 2.06
2737  KA = 5.47 * (I + P (R, S) ) – 2.06
2740  L = L + ( ( (K * KA) * ( (.18 * (P (R, S) – S, R) ) )
      2) )/K + KA) )
2745  NEXT: NEXT
2750  PRINT "L = "; L * 6.25
2755  INPUT "LO = "; LO
2760  G = (L + LO)/4
2765  PRINT "FREE ENERGY ="; G
2770  END
```

Pyridene input data

H (1, 1) = .5
H (1, 2) = 1
H (1, 3) = 0
H (1, 4) = 0
H (1, 5) = 0
H (1, 6) = 1
H (2, 2) = 0
H (2, 3) = 1
H (2, 4) = 0
H (2, 5) = 0
H (2, 6) = 0
H (3, 3) = 0
H (3, 4) = 1
H (3, 5) = 0
H (3, 6) = 0
H (4, 4) = 0
H (4, 5) = 1
H (4, 6) = 0
H (5, 5) = 0
H (5, 6) = 1
H (6, 6) = 0

NE (I) No. of electrons in the Ith M.O.
NE (1) = 2
NE (2) = 2
NE (3) = 2
NE (4) = 0
NE (5) = 0
NE (6) = 0

RESULTS

Pyridene
Eigenvalues
E (1) = – 2.10745
E (2) = – 1.16719
E (3) = – 1
E (4) = .840961
E (5) = .999999
E (6) = 1.93368

```
Charge Densities
Q (1) = 1.19521
Q (2) = .922953
Q (3) = 1.00449
Q (4) = .949912
Q (5) = 1.00449
Q (6) = .922953
Eigenvectors
u (1, 1) = .520706
u (1, 2) = .571374
u (1, 3) = 1.47859E – 07
u (1, 4) = .545913
u (1, 5) = 2.75323E – 07
u (1, 6) = – .323073
u (2, 1) = .418504
u (2, 2) = .190608
u (2, 3) = – .5
u (2, 4) = – .366024
u (2, 5) = – .5
u (2, 6) = .393128
u (3, 1) = .361268
u (3, 2) = .348896
u (3, 3) = – .5
u (3, 4) = – .238101
u (3, 5) = .499999
u (3, 6) = – .43711
u (4, 1) = .342849
u (4, 2) = .597838
u (4, 3) = – 1.42913E – 07
u (4, 3) = .566257
u (4, 5) = 2.96085E – 07
u (4, 6) = .452101
u (5, 1) = .361268
u (5, 2) = .348897
u (5, 3) = .5
u (5, 4) = – .2381
u (5, 5) = – .5
u (5, 6) = – .437109
u (6, 1) = .418504
u (6, 1) = .418504
u (6, 2) = – .190608
u (6, 3) = .5
u (6, 4) = – .366024
u (6, 5) = .5
u (6, 6) = .393127

Bondorders
P (1, 1) = 1.19521
P (1, 2) = .653652
P (1, 3) = – 2.24717E – 02
P (1, 4) = – .326131
P (1, 5) = – 2.24719E – 02
P (1, 6) = .653652
P (2, 1) = 0
P (2, 2) = .922953
P (2, 3) = .669378
P (2, 4) = 5.90611E – 02
P (2, 5) = – .330621
P (2, 6) = – .077046
P (3, 1) = 0
P (3, 2) = 0
P (3, 3) = 1.00449
P (3, 4) = .664888
P (3, 5) = 4.48656E – 03
P (3, 6) = – .330621
P (4, 1) = 0
P (4, 2) = 0
P (4, 3) = 0
P (4, 4) = .949912
P (4, 5) = .664888
P (4, 6) = .059061
P (5, 1) = 0
P (5, 2) = 0
P (5, 3) = 0
P (5, 4) = 0
P (5, 5) = 1.00449
P (5, 6) = .669378
P (6, 1) = 0
P (6, 2) = 0
P (6, 3) = 0
P (6, 4) = 0
P (6, 5) = 0
P (6, 6) = .922953
```

Acknowledgement. Financial Assistance from Pakistan Science Foundation (PSF-C-QU-Chem. 137) is gratefully acknowledged.

REFERENCES

1. See. e.g. (a) Robert G. Parr, *Quantum Theory of Molecular Electronic Structure,* W.A. Benjamin, 1963; (b) A. Streitwieser Jr., *Molecular Orbital Theory for Organic Chemists* (John Wiley, 1961), (c) M.J.S. Dewar, *The Molecular Orbital Theory of Organic Chemistry* (McGraw Hill, 1969).
2. (a) See ref. 1 (b) above; (b) L. Salem, *Molecular Orbital Theory of Conjugated System,* Benjamine 1966; (c) K. Yates, *Huckel Molecular Orbital Theory* (Academic Press, 1978); (d) C.A. Coulson, S.O. Leary and R.B. Mallion, *Huckel Theory for Organic Chemists* (Academic Press, 1978); (d) For some Recent Applications see Y. Ikegami, J. Phys. Chem., **89**, 339 (1985), M. Mohammad *et al,* Proceedings Conference on Industrial Uses of Electrochemistry, Islamabad, Pakistan, Dec. 1982.
3. H.H. Greenwood, *Computing Methods in Organic Chemistry* (Wiley, 1972).
4. H.M. Gladney, Technical Report LCAOSCF Programme, Quantum Chemistry Programme Exchange, (Indiana University, Bloomington, New Jersey, USA).
5. G.W. Wheland and D.E. Mann, J. Chem. Phys., **17**, 264 (1949).
6. See ref. 1(b) above.