

Name: Asif Iqbal Bhatti

Assignment # 2

Course Title: Computational methods of Electronic structure theory (CMEST)

Submitted to: Andris Gulans

Solution to Hartree Fock IDE via Basis sets approach

May 17, 2015

1 HF Equation: Helium atom

The atomic number of Helium atom is 2. The general Hartree-Fock equation is represented in spin orbital form by ¹

$$-\frac{\nabla_1^2}{2}\phi_k(1)-\frac{Z}{r_1}\phi_k(1)+\sum_j\int\phi_j^*(2)\frac{1}{r_{12}}[\phi_j(2)\phi(1)-\delta(\sigma_k,\sigma_j)\phi_k(2)\phi_j(1)]d\tau_2=\epsilon_k\phi_k(1) \quad (1)$$

where k is the orbital and j the sum over the number of electrons.

TASK 1-4

1. For the case of Helium atom, the above equation can be reduced by inserting $k = 1$ and varying j from 1 to 2. The resulting equation is then given by,

$$-\frac{\nabla_1^2}{2}\phi_1(1)-\frac{Z}{r_1}\phi_1(1)+\int\phi_2^*(2)\frac{1}{r_{12}}[\phi_2(2)\phi_1(1)-\delta(\sigma_1,\sigma_2)\phi_1(2)\phi_2(1)]d\tau_2=\epsilon_1\phi_1(1) \quad (2)$$

The paired electrons in the spherical symmetric spin-orbital is zero. The exchange term dissolve leaving out the Hartree part entailing the mean field approximation for Coulombic interaction.

1. With four Gaussian basis the energy is -2.855160 a.u. which is the Hartree Fock limit. Actual value is -2.903 a.u.

In above calculation following coefficient were taken,

Exponents	Ref. Jos Thijssen	Given Assignment
α_1	0.298073	6.36242139
α_2	1.242567	1.15892300
α_3	5.782948	0.31364979
α_4	38.474970	
Total Energy (a.u.)	-2.855160	-2.816246

¹Modern quantum chemistry Szabo and Ostlund

Appendix

```

1  program HF_Helium
2  use matrixmultiplication ! using module for matrix multi. & calling DSYEV
3  implicit none
4  integer                :: p, n_alpha, iter
5  integer                :: i, j, k, l, maxiter = 200
6  double precision       :: start_time, stop_time
7  double precision, parameter :: pi = 4.0*ATAN(1.0d0), Z = 2, x = 1
8  double precision       :: aa, eold, anew
9  double precision, dimension(:), allocatable :: alpha, e, c
10 double precision, dimension(:,:), allocatable :: h, f, s, v
11 double precision, dimension(:,:,:), allocatable :: q
12 character (len=100)    :: filename
13 !*****Reading parameters from file *****
14 write (*,*) "Enter the Filename for Gaussian parameters :: "
15 read (*,*) filename
16 write (*,*) "Reading from a file ... "
17 open (unit=10, file=trim(filename), status='old', action='read')
18 write (*,*) "Number of Gaussians detected ~~>> "
19 read (10,*) n_alpha
20 allocate (alpha (n_alpha))
21 do p = 1, n_alpha
22   read (10,*) alpha(p)
23   write (*,*) "For Gaussian # ",p,"Coefficient detected ~~>> ", alpha(p)
24   if ( alpha(p) <= 0.0d0 ) stop '-ve coefficient non acceptable !!! '
25 end do
26 !*****
27 !!!Inserting the Coulomb integral with the matrix elements of the e-e interaction:
28 !!!q(i,j,k,l) = Integral[ Integral{ chi_i(r) chi_j(r') 1/|r'-r| chi_k(r) chi_l(r') dr dr'} ]
29 !!!where chi_i(r) = Exponential(-alpha(i)*(r**2)) are the Gaussians
30
31 allocate ( q(n_alpha, n_alpha, n_alpha, n_alpha) )
32 do i=1,n_alpha
33   do j=1,n_alpha
34     do k=1,n_alpha
35       do l=1,n_alpha
36         q(i,j,k,l) = x * (2.0d0*pi**2.5d0)/(( alpha(i)+alpha(j))*( alpha(k)+alpha(l)) &
37           * sqrt(alpha(i)+alpha(j)+alpha(k)+alpha(l)))
38       end do
39     end do
40   end do
41 end do
42 !*****Overlap integrals S and one-electron hamiltonian H on the GB*****
43 allocate ( s(n_alpha, n_alpha), h(n_alpha, n_alpha), f(n_alpha, n_alpha), &
44 v(n_alpha, n_alpha), c(n_alpha), e (n_alpha) )
45 do i=1,n_alpha
46   do j=1,n_alpha
47     aa = alpha(i)+alpha(j)
48     s(i,j) = (pi/aa)**1.5d0
49     h(i,j) = (s(i,j)*3.0d0*alpha(i)*alpha(j))/aa - (Z*2*pi)/aa
50   end do
51 end do
52 !*****writing matrix to a file *****
53 open(unit = 555, file = 'fock.dat', status = 'replace', action = 'write')
54 do i = 1, n_alpha

```

```

55  write (555,'(F15.8)') (h(i,j), j = 1, n_alpha)
56  enddo
57  close(555)
58  !*****Starting Initial Guess for solving HF equations*****
59  do i=1,n_alpha
60    c(i) = 0.0d0
61  end do
62  c(1) = 1.1d0; c(2) = 1.50d0
63  !*****Self-consistency iteration*****
64  enew = 0.0d0
65  print*, "Coefficients initialised with a guess ..."
66  print*, "Entering Main SCF Loop ..."
67  write (*,*) "SCF #      HF-Eigenvalue      Energy      Enew-Eold"
68  call cpu_time(start_time)
69  !*****Reference:: Jos Thijssen "Comp. physic"*****
70
71  do iter = 1, maxiter
72  !*****Fill the Fock matrix*****
73    do i=1,n_alpha
74      do j=1,n_alpha
75        f(i,j) = h(i,j)
76        do k=1,n_alpha
77          do l=1,n_alpha
78            f(i,j) = f(i,j) + q(i,j,k,l) * c(k)*c(l)
79          end do
80        end do
81      end do
82    end do
83
84    !***** V(j,i), j=basis function index, i= eigenvalue index****
85
86    call diag( n_alpha, n_alpha, f, s, e, v )
87    c(:) = v(:,1)
88    eold = enew
89    enew = 0.0d0
90    do i = 1, n_alpha
91      do j = 1, n_alpha
92        enew = enew + 2.0*h(i,j)*c(i)*c(j)
93        do k = 1, n_alpha
94          do l = 1, n_alpha
95            enew = enew + q(i, j, k, l)*c(i)*c(j)*c(k)*c(l)
96          end do
97        end do
98      end do
99    end do
100    write(*, 100) iter, e(1), enew, enew-eold 100  format(2x,I4,2(6x,F10.6),6x,F15.12)
101    if ( abs (enew-eold) < 1.0d-8 ) then
102      print*, '//achar(27)//'[94m Convergence criterion satisfied exiting ... '//achar(27)//'[0m'
103      call cpu_time(stop_time)
104      print*, "Loop time", stop_time - start_time, "seconds"
105      deallocate ( e, c, v, f, h, s, alpha )      stop
106    end if
107  end do
108  print*, '//achar(27)//'[31m Convergence failed ... '//achar(27)//'[0m'
109  deallocate ( e, c, v, f, h, s, alpha )
110  stop

```

```
111 end program HF_Helium
```

```

1  module matrixmultiplication
2      implicit none
3      contains
4      subroutine diag ( n, ldh, h, s, e, v )
5          integer, intent(in)          :: n, ldh
6          double precision, intent(in)  :: h(ldh,n), s(ldh,n)
7          double precision, intent(out) :: e(n), v(ldh,n)
8          integer                      :: lwork, info, i, j, nn
9          double precision, parameter  :: small=1.d-10
10         double precision, allocatable :: work(:), b(:, :), h1(:, :)
11         info = 0
12         lwork = 3*n
13         allocate (work(lwork), b(ldh,n))
14         !*****Copy S into an auxiliary matrix because dsyev destroys the matrix
15         b = s
16         !*****Diagonalize S matrix*****
17         call dsyev ( 'V', 'U', n, b, ldh, e, work, lwork, info )
18         if (info /= 0) stop 'S-matrix diagonalization failed '
19         !***Keep only linearly independent combinations (within a given threshold)
20         !***store into matrix "b" the eigenvectors of S divided by the squares of the eigenvalues
21         nn = 0
22         do i=1,n
23             if (e(i) > small) then
24                 nn = nn + 1
25                 b(:,nn) = b(:,i) / sqrt(e(i))
26             end if
27         !*****print *, aux
28         end do
29         if ( nn < n ) write(*,*) " # of linearly independent vectors =", nn, n
30         !*****Transform H using the "B" matrix*****
31         !***** $V(i,j) = \sum_{k=1}^n \{H(i,k) \times B(k,j)\}$ ,  $i=1,n, j=1,nn$ 
32
33         call dgemm ( 'N', 'N', n, nn, n, 1.0d0, h, ldh, b, ldh, 0.0d0, v, ldh )
34         !***** $h1(i,j) = \sum_{k=1}^n \{B(k,i) \times v(k,j)\}$ ,  $i=1,nn, j=1,nn$ 
35         !***** $H' = b^T H b$ *****
36
37         allocate (h1(nn,nn) )
38         call dgemm ( 'T', 'N', nn, nn, n, 1.0d0, b, ldh, v, ldh, 0.0d0, h1, nn )
39         !*****Diagonalize H again*****
40
41         info = 0
42         call dsyev ( 'V', 'U', nn, h1, nn, e, work, lwork, info )
43         if (info /= 0) stop 'H-matrix diagonalization failed '
44         !*****Back-transform eigenvectors*****
45
46         call dgemm ( 'N', 'N', n, nn, nn, 1.0d0, b, ldh, h1, nn, 0.0d0, v, ldh )
47         deallocate (h1, b, work)
48         end subroutine diag
49
50 end module matrixmultiplication

```