```matlab
%------------------------------------------------------------
% Tony Hyun Kim
% 8.06: Hartree-Fock implementation on HeH+ ground state
%       using minimal 1s STO-3G basis set.
%
% General notes:
%   - All quantities are expressed in atomic units
%------------------------------------------------------------

dP = 1E-6;    % Convergence criterion using the std dev. of density matrix
maxIter = 50; % Maximum number of iterations before we force quit

x = linspace(-2,3,500); % The range of x over which the iterative
                        % wavefunctions will be drawn.

%------------------------------------------------------------
% STEP 1:
%------------------------------------------------------------
% First, we specify the molecule (HeH+). We will use index 1
% to refer to He, fixed at the origin; and use index 2 to
% refer to H, located at [R 0 0].
%------------------------------------------------------------
Z = [2 1];
R = 1.4632;

% Two electrons
N = 2;

%------------------------------------------------------------
% CHOOSING A BASIS SET
%------------------------------------------------------------
% STO-3G parameters. We are using a minimal basis set of two
% 1s-type orbitals. Each function consists of three primitive
% gaussians of the form (2*a(k)/pi)^(3/4)*exp(-a(k)*(r-R)^2)
% weighted by the coefficient d(k)
%------------------------------------------------------------
d = [0.444635 0.535328 0.154329];
a = [0.109818 0.405771  2.22766]; % Coefficients for zeta = 1.0

% Scale a to fit zeta != 1
zeta_he = 2.0925;
zeta_h  = 1.24;
a_he = zeta_he^2*a;
a_h  = zeta_h^2*a;

a_prim = [a_he a_h];
R_prim = [0 0 0 R R R];

%------------------------------------------------------------
% STEP 2:
%------------------------------------------------------------
% The following matrices need to be computed only ONCE per a
% set of iterations.
%
% Compute S:  the overlap matrix,
%         T:  the kinetic energy matrix,
%         V1: the nuclear potential due to He
%         V2: the nuclear potential due to H
%         H:  core-Hamiltonian, H = T + V1 + V2
%
% Also compute the two electron integrals I. (The most time
% consuming operation.)
%------------------------------------------------------------
S  = zeros(2,2);
T  = zeros(2,2);
V1 = zeros(2,2);
V2 = zeros(2,2);
H  = zeros(2,2);
for i = 1:2
    % Note that we only compute the upper triangle of each of these
    % matrices. The other half is specified by symmetry.
    for j = i:2
        % The calculations are easily done with the primitive
        % Gaussians that make up our basis functions.
        for p = 1:3
```

```
            for q = 1:3
                % Intermediate variables
                a1  = a_prim(3*(i-1)+p);
                a2  = a_prim(3*(j-1)+q);
                as  = a1 + a2;
                ap  = a1 * a2;
                rat = ap/as;
                % Displacement of centers of the primitive Gaussians
                R1 = R_prim(3*(i-1)+p);
                R2 = R_prim(3*(j-1)+q);
                disp = R1 - R2;
                % Location of the product gaussian
                Rp = (a1*R1+a2*R2)/(a1+a2);

                %----------------------------------------------------------
                % Overlap matrix, S(i,j)
                %----------------------------------------------------------
                s_pq = 2^(3/2)*(rat/as)^(3/4)*exp(-rat*disp^2);
                S(i,j) = S(i,j) + d(p)*d(q)*s_pq;

                %----------------------------------------------------------
                % Kinetic energy matrix, T(i,j)
                %----------------------------------------------------------
                T(i,j) = T(i,j) + d(p)*d(q)*rat*(3-2*rat*disp^2)*s_pq;

                %----------------------------------------------------------
                % Nuclear potential of Helium, V1(i,j), refers to a helper
                % function F0(t) = 1/2*sqrt(pi/t)*erf(sqrt(t)). We define
                % F(0) = 1.
                %----------------------------------------------------------
                V1(i,j) = V1(i,j) + d(p)*d(q)*(2/pi)^(3/2)*ap^(3/4)*-2*pi/as*Z(1)*exp(-rat*disp^2)*F0(as*Rp^2);

                %----------------------------------------------------------
                % Nuclear potential of Hydrogen, V2(i,j)
                %----------------------------------------------------------
                V2(i,j) = V2(i,j) + d(p)*d(q)*(2/pi)^(3/2)*ap^(3/4)*-2*pi/as*Z(2)*exp(-rat*disp^2)*F0(as*(Rp-R)↙
^2);
            end
        end

        % By symmetry
        S(j,i)  = S(i,j);
        T(j,i)  = T(i,j);
        V1(j,i) = V1(i,j);
        V2(j,i) = V2(i,j);
    end
end

% Assign the core-Hamiltonian
H = T + V1 + V2;

%----------------------------------------------------------
% Now we compute the two electron integrals, I(i,j,k,l)
%
% Since each index can select between our two basis functions,
% we must in principle fill out 2^4 = 16 entries. However,
% because our basis functions are real, there is an eight-fold
% symmetry in the two-electron integrals.
%----------------------------------------------------------
I = zeros(2,2,2,2);
for i = 1:2
    for j = i:2
        for k = i:2
            for l = k:2
                %----------------------------------------------------------
                % START inner loop:
                %
                % I apologize for the extreme nesting of loops. But
                % we're going to compute the overlaps with the primitive
                % Gaussians that make up the basis functions.
                %----------------------------------------------------------
                I(i,j,k,l) = 0;
                for p = 1:3
                    for q = 1:3
                        for r = 1:3
```

```
                            for s = 1:3
                                a1 = a_prim(3*(i-1)+p);
                                a2 = a_prim(3*(j-1)+q);
                                a3 = a_prim(3*(k-1)+r);
                                a4 = a_prim(3*(l-1)+s);

                                as1 = a1 + a3;
                                as2 = a2 + a4;
                                as  = as1 + as2;
                                ap  = a1*a2*a3*a4;
                                rat1 = a1*a3/as1;
                                rat2 = a2*a4/as2;

                                R1 = R_prim(3*(i-1)+p);
                                R2 = R_prim(3*(j-1)+q);
                                R3 = R_prim(3*(k-1)+r);
                                R4 = R_prim(3*(l-1)+s);

                                Rp = (a1*R1+a3*R3)/as1;
                                Rq = (a2*R2+a4*R4)/as2;

                                disp1 = R1-R3;
                                disp2 = R2-R4;
                                disp  = Rp-Rq;

                                I(i,j,k,l) = I(i,j,k,l) + d(p)*d(q)*d(r)*d(s)*16/sqrt(pi)*ap^(3/4)/ ↙
(as1*as2*sqrt(as))*exp(-rat1*disp1^2-rat2*disp2^2)*F0(as1*as2/as*disp^2);
                            end
                        end
                    end
                end
                %----------------------------------------------------------
                % END inner loop
                %----------------------------------------------------------

                % We apply the symmetry of I
                I(i,l,k,j) = I(i,j,k,l);
                I(j,i,l,k) = I(i,j,k,l);
                I(j,k,l,i) = I(i,j,k,l);
                I(k,j,i,l) = I(i,j,k,l);
                I(k,l,i,j) = I(i,j,k,l);
                I(l,i,j,k) = I(i,j,k,l);
                I(l,k,j,i) = I(i,j,k,l);
            end
        end
    end
end


%------------------------------------------------------------
% STEP 3:
%------------------------------------------------------------
% - Diagonalize the overlap matrix S.
% - Obtain the transformation matrix X (symmetric orthogonalization)
%------------------------------------------------------------
X = zeros(2,2);
s_invroot = zeros(2,2);

% The function 'eig' gives the eigenvectors in columns of U. This is also
% the unitary matrix U s.t. U'*S*U is diagonal. (i.e. U'*S*U = s)
[U,s] = eig(S);
for i = 1:2
    s_invroot(i,i) = s(i,i)^(-1/2);
end
X = U*s_invroot*U';

%------------------------------------------------------------
% STEP 4:
%------------------------------------------------------------
% - Guess the density matrix P
%------------------------------------------------------------
P = [0 0; 0 0];

%------------------------------------------------------------
% STEP 5:
```

```
%-------------------------------------------------------------
% We now ENTER THE SCF LOOP
%
% - Compute the matrix G from P and the two-electron integrals.
% - Compute the Fock matrix F = H_core + G;
%-------------------------------------------------------------
F = zeros(2,2);

% Loop variables
count = 0;
sigma = 1;
clear disp;

% Enter SCF loop
while(sigma > dP)
    P_prev = P;
    count = count + 1;
    disp('Step #');
    disp(count);

    %-------------------------------------------------------------
    % Construct the matrix G
    %-------------------------------------------------------------
    G = zeros(2,2);
    for i = 1:2
        for j = 1:2
            % BEGIN inner loop
            for p = 1:2
                for q = 1:2
                    G(i,j) = G(i,j) + P(p,q)*(I(i,q,j,p)-1/2*I(i,q,p,j));
                end
            end
            % END inner loop
        end
    end

    %-------------------------------------------------------------
    % STEP 6:
    %-------------------------------------------------------------
    % Construct the Fock matrix F
    %-------------------------------------------------------------
    F = H + G;

    %-------------------------------------------------------------
    % STEP 7:
    %-------------------------------------------------------------
    % Obtain the transformed Fock matrix Fp = X'*F*X
    %-------------------------------------------------------------
    Fp = X'*F*X;

    %-------------------------------------------------------------
    % STEP 8:
    %-------------------------------------------------------------
    % Diagonalize Fp to obtain C' and eps
    %-------------------------------------------------------------
    [Cp, eps] = eig(Fp);

    %-------------------------------------------------------------
    % STEP 9:
    %-------------------------------------------------------------
    % Transform Cp to the C in F*C=S*C*eps
    %-------------------------------------------------------------
    C = X*Cp;

    %-------------------------------------------------------------
    % STEP 10:
    %-------------------------------------------------------------
    % Form a new density matrix P
    %-------------------------------------------------------------
    P = zeros(2,2);
    for i = 1:2
        for j = 1:2
            % BEGIN inner loop
            for k = 1:(N/2)
                P(i,j) = P(i,j) + 2*C(i,k)*C(j,k);
```

```
            end
            % END inner loop
        end
    end

    %------------------------------------------------------------
    % STEP 11:
    %------------------------------------------------------------
    % Determine whether the procedure has converged. We will monitor
    % the standard deviation of the successive density matrix
    % elements for this purpose.
    %------------------------------------------------------------
    sigma = 0;
    for i = 1:2
        for j = 1:2
            sigma = sigma + (P(i,j)-P_prev(i,j))^2;
        end
    end
    sigma = (1/4*sigma)^(1/2);

    %------------------------------------------------------------
    % Output results: draw the density of a single electron
    %                 over a cross-section (x,0,0)
    %------------------------------------------------------------
    psi = zeros(1,length(x));
    for i = 1:2
        % Each basis function is built with three Gaussians
        for p = 1:3
            a1  = a_prim(3*(i-1)+p);
            R1  = R_prim(3*(i-1)+p);
            psi = psi + C(i,1)*d(p)*(2*a1/pi)^(3/4)*exp(-a1*(x-R1).^2);
        end
    end
    plot(x,psi.^2);
    hold on;

    %------------------------------------------------------------
    % A brake mechanism. Force-quit if we exceed the specified
    %   maximum iteration.
    %------------------------------------------------------------
    if (count >= maxIter)
        sigma = 0;
    end
end

%------------------------------------------------------------
% Final output
%------------------------------------------------------------
grid on;
plot(x,psi.^2,'r','LineWidth',2);
xlabel('x in a.u.');
ylabel('Density of a single electron through a cross-section (x,0,0)');
title('Ground state wavefunction for HeH+. Red indicates the converged solution');
```