HOMEWORK 3

Hartree-Fock Methods

Due: Monday February 16, 4:30 pm

Problem 1: Symmetries in Hartree-Fock

Suppose you are performing a Hartree-Fock calculation using the finite basis $\chi_p(r)$.
Remember that our HF formulation looks like this:

$$FC_k = \epsilon SC_k$$

where **F** is given as:

$$F_{pq} = h_{pq} + \sum_k \sum_{rs} C^*_{rk} C_{sk} (2\langle pr|g|qs\rangle - \langle pr|g|sq\rangle)$$

with

$$h_{pq} = \langle p|h|q\rangle = \int d^3r \chi^*_p(r)\left[-\frac{1}{2}\nabla^2 - \sum_n \frac{Z_n}{|R_n - r|}\right]\chi_q(r)$$

and

$$\langle pq|g|rs\rangle = \int d^3r_1 d^3r_2 \chi^*_p(r_1)\chi^*_r(r_2)\frac{1}{|r_1 - r_2|}\chi_q(r_1)\chi_s(r_2)$$

The kinetic and Coulomb integrals must be calculated. We can gain speed by using the symmetry of the matrices, for example:

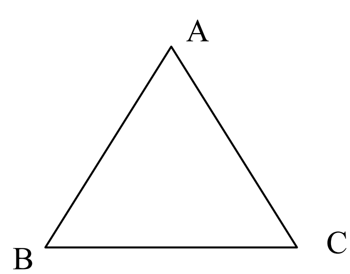
$$\langle \chi_p|\nabla^2|\chi_q\rangle = \langle \chi_q|\nabla^2|\chi_p\rangle$$

Assume that there are no other symmetries present in our system.

a) If our basis contains M basis functions, at least how many of these matrix elements must be calculated?

b) For the two electron matrix elements, $\langle pq|g|rs\rangle$, how many elements must be calculated?

Now suppose the molecule for which we are performing the calculation consists of three identical atoms (A, B, and C), located on an equilateral triangle. On every atom, we have M basis functions denoted by $\chi^A_p(r)$ with p=1,…,M.



The basis functions on the different atoms have the form:

$$\chi_p^A(r) = \chi_p(r - R_A),$$
$$\chi_p^B(r) = \chi_p(r - R_B), etc...$$

c) How many different kinetic and coulomb matrix elements must be calculated in this case?
d) How many different two-electron matrix elements must be calculated?
e) Suppose M is very large, so that you are using a big basis. What is then the gain in speed when using all symmetries of the matrix elements instead of using no symmetry at all?

Problem 2: A Hartree-Fock program for calculating the helium ground state energy

Let's assume that the total spin of the electrons in the He atom is zero. Writing the total wave function as a direct product of two single-electron functions $\phi(r)$, we can derive the Schrödinger equation for an electron as follows:

$$\left[\frac{1}{2}\nabla^2 + E - V(r)\right]\phi(r) = 0$$

where V(r) is the potential energy of an electron moving in the field of the nucleus and electron charge:

$$V(r) = -\frac{2}{r} + \int \frac{1}{r_{12}} |\phi(r_2)|^2 \, d^3 r_2$$

We will solve this nonlinear equation by representing ϕ as an expansion over four Gaussian functions:

$$\phi(r) = \sum_{p=1}^{4} C_p \chi_p(r),$$
$$\chi_p(r) = \exp(-\alpha_p r^2)$$
$$\alpha_1 = 0.297104$$
$$\alpha_2 = 1.236745$$
$$\alpha_3 = 5.749982$$
$$\alpha_4 = 38.216677$$

For this basis set, the Schrodinger equation can be represented in matrix form:

$$\sum_q \left(h_{pq} + \sum_{rs} C_r C_s Q_{prqs} \right) C_q = E' \sum_q S_{pq} C_q$$

where

$$h_{pq} = \int d^3 r \chi_p^*(\mathbf{r}) \left[-\frac{1}{2}\nabla^2 - \frac{2}{\mathbf{r}} \right] \chi_\mathbf{q}(\mathbf{r}),$$
$$Q_{prqs} = \int\int d^3 r_1 d^3 r_2 \chi_p^*(\mathbf{r_1}) \chi_\mathbf{r}^*(\mathbf{r_2}) \frac{1}{|\mathbf{r_i} - \mathbf{r_j}|} \chi_\mathbf{q}(\mathbf{r_1}) \chi_\mathbf{s}(\mathbf{r_2}),$$
$$S_{pq} = \int d^3 r \chi_p^*(\mathbf{r}) \chi_\mathbf{q}(\mathbf{r}),$$

Since we are using simple Gaussians, these integrals can be computed analytically:

$$S_{pq} = \left[\frac{\pi}{\alpha_p + \alpha_q}\right]^{3/2},$$

$$h_{pq} = 3\pi^{3/2}\frac{\alpha_p\alpha_q}{(\alpha_p+\alpha_q)^{5/2}} - \frac{4\pi}{\alpha_p+\alpha_q},$$

$$Q_{prqs} = \frac{2\pi^{5/2}}{(\alpha_p+\alpha_q)(\alpha_r+\alpha_s)\sqrt{\alpha_p+\alpha_q+\alpha_r+\alpha_s}}.$$

Calculate the helium ground state energy by following these steps:

a) Calculate and fill all the matrices

b) Choose initial values for C_p

c) Use the values to construct the Fock matrix,F: (the matrix corresponding to the Hamiltonian in our matrix formulation of the Schrödinger equation. Remember, the vector C always needs to be normalized before plugging it into the Schrödinger equation (**be careful normalizing, we are not using an orthonormal basis**).

d) Solve the generalized eigenvalue problem **FC**=E**SC.** We are interested in the eigenvector corresponding to the lowest eigenvalue.

e) Organize the iteration loop

f) When the convergence is reached, calculate the total ground state energy

$$E_g = 2\sum_{pq} C_p C_q h_{pq} + \sum_{pqrs} Q_{prqs} C_p C_r C_q C_s$$

g) what do you get if you use this different formula?

$$E_g = 2E_G^1 - \sum_{pqrs} Q_{prqs} C_p C_r C_q C_s$$

where E_G^1 is the lowest single-particle energy.

Submit both the source code and a table of the total energy calculated by the two formulas as a function of iteration number (do not show more than the first 20 iterations)

For problem 2, you will need to be able to solve a generalized eigenvalue problem. If you are using C++ you might want to look into GSL. Matlab also has the ability to solve generalized eigenvalue problems. Details about the implementation follows below:

Implementation of GSL on Windows

If you are using Dev-C++:
http://www.quantcode.com/modules/smartfaq/faq.php?faqid=10

If you are using the Microsoft C++ product:
http://www.quantcode.com/modules/smartfaq/faq.php?faqid=33

Solving Generalized Eigenvalue Problems

If you are using Matlab:

d = eig(A,B) returns a vector containing the generalized eigenvalues, if A and B are square matrices. For the generalized eigenvalue problem $Ax=\lambda Bx$
[V,D] = eig(A,B) produces a diagonal matrix D of generalized eigenvalues and a full matrix V whose columns are the corresponding eigenvectors so that A*V = B*V*D .
More information can be found here: link

If you are using GSL:
In GSL generalized eigenvalue problems are formulated as: $\beta Ax = \alpha Bx$
Where the normal eigenvalues $\lambda = \alpha / \beta$
Implementation of the generalized eigenvalue solver is given by these functions:
http://www.gnu.org/software/gsl/manual/html_node/Real-Generalized-Nonsymmetric-Eigensystems.html
Keep in mind that your eigenvalue problem may include, real, symmetric, sparse, or hermitian matrices which may make your problem computationaly easier by using some of these functions:
http://www.gnu.org/software/gsl/manual/html_node/Eigensystems.html