Project 1: Make your own Hartree-Fock program

March 1, 2019

In this project you will build your own Hartree-Fock program using the infrastructure provided by the DIRAC electronic structure program.

I. Theory

In Hartree-Fock theory we minimize the energy of a many-electron system by assuming a single determinant form of the wave function. For derivation of the equations and further background we refer to the course Understanding Quantum Chemistry that we assume to be followed prior to taking this project assignment. We will below just summarize the working equations that are to be programmed in the project. For simplicity we assume that we work with a basis of n orthogonal spinorbitals so that the overlap matrix need not be considered explicitly.

We start by defining a density matrix $\mathbf{D}$ that is constructed from the matrix of MO coefficients $\mathbf{C}$ as

$$D_{pq} = \sum_{i}^{occupied} C_{pi}^{*} C_{qi} \tag{1}$$

With this density matrix we can compute the Fock matrix $\mathbf{F}$ as

$$F_{pq} = h_{pq}^{core} + \sum_{r,s}^{occupied} g_{prqs} D_{rs}; \tag{2}$$

and the Hartree-Fock energy E^{HF} as

$$E^{HF} = \sum_{p,q} h_{pq}^{core} D_{pq} + \frac{1}{2} \sum_{p,q,r,s}^{occupied} g_{prqs} D_{pq} D_{rs}; \tag{3}$$

if both the set of 1-electron $\mathbf{h}^{core}$ and anti-symmetrized 2-electron $\mathbf{g}$

$$g_{prqs} = \int\int \psi_p^*(1)\psi_r^*(2)\frac{1}{r_{12}}(1 - P_{12})\psi_q(1)\psi_s(2) d\mathbf{x}_1 d\mathbf{x}_2; \tag{4}$$

integrals are available. If we do not consider frozen cores $\mathbf{h}^{core}$ is simply the matrix representation of the sum of the kinetic energy and the potential energy operators. If we allow for a frozen core this matrix also includes the interaction with a set of predetermined core orbitals.

The simplest Hartree-Fock optimization that can be programmed consists of four steps

1. Calculate trial density
2. Construct the Fock matrix
3. Diagonalize the Fock matrix to obtain n eigenvalues $\{\epsilon\}$ and eigenvectors $\{\mathbf{C}\}$
4. Check convergence and if not converged repeat step 2-4
5. Upon convergence (or if a maximum number of iterations is reached): Calculate E^{HF}

We will discuss these steps below in a bit more detail.

II. Implementation

II.1. Calculate trial density

Since we will use the molecular orbitals generated by the restricted SCF formalism implemented in DIRAC, a diagonal matrix will provide a good start guess. The main task will be to read input from the user of your program to specify how many spinors need to be occupied. Goal is to be able to also perform unrestricted calculations, but you can start to test your program with a restricted, closed shell, calculation in which a diagonal trial density should immediately give a converged result.

II.2. Construct the Fock matrix

The main challenge in this part is to get familiar with the infrastructure of the ExaCorr module that uses the TALSH library developed by Lyakh. This is a finished part of a larger development (the ExaTensor library) aimed at massively parallel computers equipped with many powerful GPUs. With TALSH we use only one node and employ the CPU rather than the GPU, which makes it suitable for use on a laptop computer. You will be provided with two tensors that contain $\mathbf{h}^{core}$ and $\mathbf{g}$, and need to construct tensors to hold the Fock and density matrices. Here the task will be to look at the documentation of TALSH and the examples provided elsewhere in the code to see how you can assign values from a regular (allocatable) Fortran array to this tensor type. After having done this the implementation of (2) and (3) can be done in a few lines of code.

II.3. Diagonalize the Fock matrix

You first need to copy the Fock matrix from the tensor format to a standardized Fortran form so that you can call a standard diagonalization routine. Technically this extraction of information from a tensor is similar to copying it into the tensor (see previous section), in both cases you use a pointer to assign the memory occupied by the data inside the tensor to an allocatable Fotran array. For diagonalizing the Fock matrix, you can call the qdiag90 routine that is present in the *src/gp/qalgebra.F* file in DIRAC. Look up this file and read the documentation of qdiag (the older version of this routine in which work memory had to be supplied explicitly) to understand how the qdiag90 around qdiag works.

II.4. Check convergence and possibly repeat the steps

The easiest check is to compare the energy calculated with that of the previous iteration and stop if the difference between these values is below a treshold (10^{-9} Hartree would be a good value for this). More robust is to check whether the density matrix is converged by either taking the vector norm of the difference in density matrices of two iterations or (and this is used in most implementations) by taking the norm of the commutator of the Fock and density matrices.

$$\epsilon^n = \sqrt{\sum_{p,q} ([\mathbf{F}^n, \mathbf{D}^{n-1}])^2_{p,q}} \tag{5}$$

For the latter check you only need one density matrix so there are fewer arrays to allocate. Note that you should hereby take the density matrix used to build the Fock matrix, not the one that you construct from the new eigenvectors.

If the check reveals that you are not converged you should construct a new density matrix from the occupied spinors and continue the SCF iterations.

II.5. Calculate Energy

If you already did this during the iterations you can simply print this once more in a nice form and provide some information about the number of steps needed to converge, convergence reached etc.

III. Extensions

The unrestricted Hartree-Fock formalism is just one of the possible self-consistent field procedures. It could be interesting to also consider state-specific optimization (as described in the Roothaan paper[2]) and/or use of convergence accelators like the DIIS method. A useful extension is also to implement the diagonalization-free algorithm of [1] as this can be implemented without having to copy out data from the tensor to the regular format.

References

[1] Marcela Hrdá, Tomáš Kulich, Michal Repisky, Jozef Noga, Olga L Malkina, and Vladimir G Malkin. Implementation of the diagonalization-free algorithm in the self-consistent field procedure within the four-component relativistic scheme. *Journal of Computational Chemistry*, 35(23):1725–1737, July 2014.

[2] C. C. J. Roothaan. Self-Consistent Field theory for open shells of electronic systems. *Reviews Of Modern Physics*, 32(2):179–185, January 1960.